Rochester Institute of Technology
RIT Scholar Works

Theses | Thesis/Dissertation Collections

2006

A biologically plausible system for detecting saliency in video

David Burlone

Follow this and additional works at: http://scholarworks.rit.edu/theses

Recommended Citation
Burlone, David, "A biologically plausible system for detecting saliency in video" (2006). Thesis. Rochester Institute of Technology. Accessed from

This Thesis is brought to you for free and open access by the Thesis/Dissertation Collections at RIT Scholar Works. It has been accepted for inclusion in Theses by an authorized administrator of RIT Scholar Works. For more information, please contact ritscholarworks@rit.edu.

A Biologically Plausible System For Detecting Saliency In Video

by

David J. Burlone

A Thesis Submitted in Fulfillment of the Requirements for the Degree of Master of Science in Computer Science

Supervised by
Director, Laboratory of Intelligent Systems Dr. Roger S. Gaborski
Department of Computer Science
Golisano College of Computing and Information Sciences
Rochester Institute of Technology
Rochester, New York
May 2006

Approved By:

Dr. Roger S. Gaborski
Director, Laboratory of Intelligent Systems
Chairman

Dr. Carl H. Reynolds
Visiting Professor
Reader

Dr. Edith Hemaspaandra
Professor
Observer

Abstract

Neuroscientists and cognitive scientists credit the dorsal and ventral pathways for the capability of detecting both still salient and motion salient objects. In this work, a framework is developed to explore potential models of still and motion saliency and is an extension of the original VENUS system. The early visual pathway is modeled by using Independent Component Analysis to learn a set of Gabor-like receptive fields similar to those found in the mammalian visual pathway. These spatial receptive fields form a set of 2D basis feature matrices, which are used to decompose complex visual stimuli into their spatial components. A still saliency map is formed by combining the outputs of convoluting the learned spatial receptive fields with the input stimuli.

The dorsal pathway is primarily focused on motion-based information. In this framework, the model uses simple motion segmentation and tracking algorithms to create a statistical model of the motion and color-related information in video streams. A key feature of the human visual system is the ability to detect novelty. This framework uses a set of Gaussian distributions to model color and motion. When a unique event is detected, Gaussian distributions are created and the event is declared novel. The next time a similar event is detected the framework is able to determine that the event is not novel based on the previously created distributions. A forgetting term is also included that allows events that have not been detected for a long period of time to be forgotten.

Acknowledgements

I would first like to thank my committee individually; Dr. Roger Gaborski for his amazingly sharp sense of humor, and technical insight; Dr. Carl Reynolds for his guidance through the thesis process; and Dr. Edith Hemaspaandra for her constant encouragement. I next want to thank my colleagues who have helped me professionally, technically, and personally: Dave, Tom, Jon, Raja, Lomax, Jim, Karthik, and the DB guys, thank you all. Finally I want to thank my mother for understanding why this was all worth it.

Contents

I Introduction

The evolution of technology has in some ways cursed end users with an overload of information. This has created new problems, and along with these new problems new solutions that root themselves in areas such as data mining most specifically in the area of novelty detection. The basic idea of novelty detection is to find information in specific regions of video, audio, or simply a still image, that is "interesting" enough to draw the human focus of attention toward. This is a task which seems completely arbitrary for a human to do, but designing algorithmic approach to perform this task is quite a different story. There have been many different hypotheses as to how the human brain is able to detect these areas of interest. Though there are several direct applications of a system which can perform the task of novelty detection the primary focus will be on the application to a security system and for the task of information reduction.

The mammalian visual pathway has been thoroughly researched over the course of decades with research continuing to the present day. This research has classified different parts of the visual pathway into sections with reasonable confidence of their primary functionality. There have been many past experiments which have deduced, based on metabolism rates in the brain, or pulse train nerve responses, how the mammalian visual pathway responds to different forms of input. Due to the vast amount of biological investigations performed, many researchers have used this to their advantage in creating systems which to some extent model the human brain. Many biologically inspired systems try to model simple cell receptors, use neural networks in one way or another, apply some form of higher level statistics, and many times employ a combination of these and other tools in an attempt to reproduce human cortical activities. Biologically motivated systems have a clear

advantage over other types of systems, because they are being modeled after a system that works quite well. The downfall of these systems is the lack of complete understanding of the cortical areas of the brain. Although there has been a vast amount of research conducted, there are still gray areas as to how visual pathway processes its input.

Much of the interest which revolves around a biologically inspired system is their ability to adapt to a given environment. Traditional computer vision and machine learning systems implemented for industrial applications are very good at only one specific task, which implies that their input domain must be under tight constraints. Welding robots, for example, may be monitored by a visual welding inspection device. Such a system can be part of a closed loop control system where based on specific facets of a weld, the system can adjust certain parameters to produce better welds. The same system can also take a more passive approach and flag welds which are of questionable quality, as compared to the rest of the welded items passing. Unfortunately this type of system is only good for evaluating the quality of a weld. A trained human can perform these same types of tasks, but has the potential to learn the qualities of a weld quicker than a machine would, along with be more accurate. The comparison between a computer system's ability to perform a specialized task and a human's ability to perform the same specialized task is quite deliberate. The common ground between these two examples is the ability to learn or be trained.

The focus of this thesis will be on the implementation and exploration of a biologically motivated system that detects salient regions spatially and temporally in video streams. Color difference channels as described by Itti [8] and orientation sensitive receptors are learned by leveraging a statistical device known as Independent Component Analysis [1][16]. These receptors root themselves into the

understanding of objects. These low level receptors would tend to be found in the early visual system. The second pathway, known as the parietal pathway, deals more with motion processing and understanding. In the implementation of this thesis, the motion processing and understanding deals primarily with generating a statistical model of the saliency of consecutive frames by creating a pool of Gaussians.

II Background

The human visual pathway is composed of different known layers of activity. These layers have been hypothesized to be made up of collections of cells, which all perform similar functions in cooperation with one another. The cells in the very lowest layer of the visual pathway are connected almost directly to the rods and cones of the retina. These cells all have a specific function, which is simply to decompose the eye's photoreceptors input into low level basic properties. These basic properties are usually very simple and can be thought of as features of input, for example intensity. As the input gets fed forward through the visual pathway, larger cells process more complex information such as edge information and patterns.

Laurent Itti *et al.* focuses their research on modeling visual attention in both top-down and bottom-up methods [8]. Bottom-up attention is basically directing attention to low level features such as areas of high intensity or areas or prominent contrast. Top-down attention is focused mostly toward the individual's outcome expectations [18]. Some works have tightly coupled top-down saliency with bottom-up saliency [4][8][18]. The work of Itti and Vaingankar has, independently, been compared side by side with human eye-tracking data to compare the accuracy and plausibility that their technique is practical.

Systems implemented by Itti, Gaborski, and Vaingankar use multiple scaled spatial filters which were simply defined as Gabor Functions or a Difference of Gaussians [4][8][18]. Although these systems yield results as to how to set a focus of attention in an image (or in some case video stream), there is one key factor which does not seem to come up. The element of learning has been taken away from these systems. Several investigators have found that over time and prolonged exposure, different receptors will adjust to different conditions and responses [2][15].

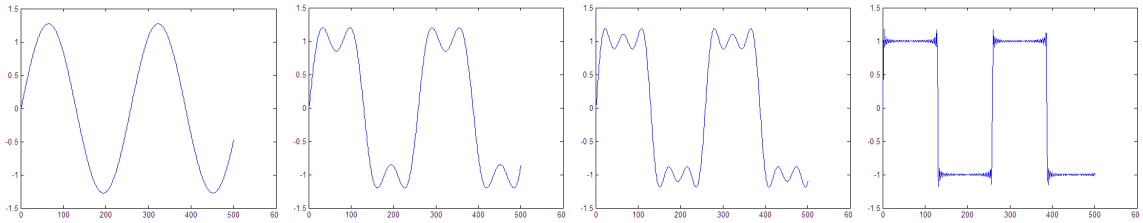
The remainder of this section will primarily focus on the tools required for generating the bottom-up saliency model of the early visual system. This section will also explain basis functions and their application to images and image processing.

2.1 Basis Functions

2.1.1 What Are Basis Functions & Their Importance In Computer Vision?

Basis Functions mathematically parallel Fourier Summations. Fourier had proposed that any complex wave could be created by summing an infinite number of sinusoidal waves of different amplitudes and phase shifts. For purposes in practical applications, only approximations are used to perform this sort of reconstruction, by only using the first N waveforms. There are countless numbers of waveforms which can be created in the fashion of adding together enough "correct" sinusoidal waves.

$$x(t) = A_0 + \sum_{k=1}^{\infty} A_k \cdot \cos(k\omega_0 t + \phi_k)$$



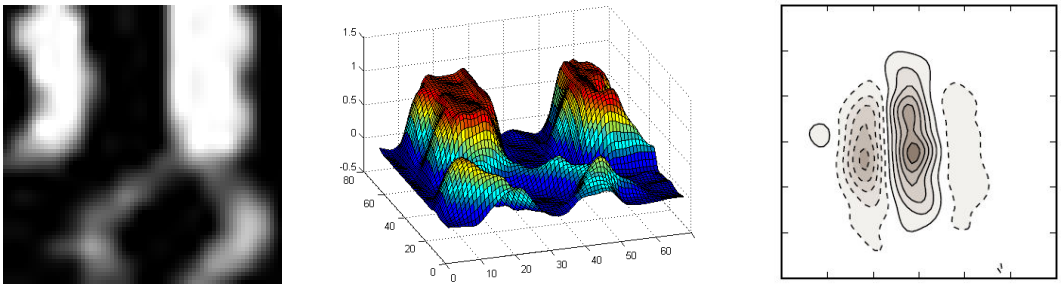
(a) (b) (c) (d)

Figure 1: Constructing a Signals Using Fourier Summation (Top). Using the Fourier Series to Partially Reconstruct a Square Wave Using Components of Sinusoidal Wave (a)n = 1 (b)n = 3 (c)n = 5 (d)n = 61 (Bottom).

A basis function is very similar, but rather than being composed strictly of sinusoidal functions with an associated phase shift and amplitude, these functions can be somewhat arbitrary, so long as they can be combined in a constructive manner to generate the original signal from which they were derived. It has been found by many researchers that the V1 layer of the visual cortex consists of many simple receptive fields, which exhibit Gabor-like features. These simple receptive cells have

also been hypothesized to be learned and developed over time, with prolonged exposure to visual elements. Work originally performed by Blakemore and Van Sluyters has experimentally shown that the receptors in young kittens develop differently when placed in visually controlled environments [2]. Basically, by placing a young kitten in environments with an abundance of objects that contain strong horizontal features, the kitten will develop strong horizontal receptors, but very few maps organized to respond to vertical input and vice versa.

If these receptive fields were to be programmed into a computer with some sensible format, when applied to an image via a sliding function (such as convolution) all have a particular response to the distinctive features in a given scene. These distinctive features, previously mentioned, are intensity, contrast, edges, edge orientation, etc. By computationally creating these receptors it's conceivable to model the early visual system.



(a) *(b)* *(c)*

Figure 2: *(a)An Example of a Receptive Field That Responds Highly To Vertical Features; (b)The Three Dimensional Representation of the Filter; (c)The Receptive Field Response of a Kitten - figure from [*15*]*

Figure 2 shows a good example of the similarity between a physical receptor *(c)* and a receptor which has been learned via the implementation to be discussed in 2.1.2. *Figure 2a* and *Figure 2b* are equivalent, just different representations of one another to help visualize this type of receptor. *Figure 2c* is a representation generated by Ohzawa, DeAngelis, and Freeman by performing reverse correlation [15]. This

representation is similar in that it evaluates the cell activities over time and then stacks them all together, and adds them. For their experimentation all of their plots lined the receptive fields up vertically, but this particular receptor was originally oriented at 100°. The artificially generated filter is one which responds highly to vertical edges within images.

2.1.2 Constructing Basis Functions

The basis function most commonly associated with the early visual system is the Gabor function. Gabor filters have been successfully used by many different systems which model the visual pathway. In many systems these filters are computed mathematically without any sort of training with real-word data input. The model basis functions these mathematical representations produce is very clean, and unlike what is found in the mammalian visual cortex.

The visual pathway is developed over time, similar to the kitten experiment as described in 2.1.1. As one is exposed to more visual input, over time one's receptive fields change and develop. Rather than using a completely computational approach to generate these receptive fields, it makes more sense to emulate the biological system. Emulating a biological system introduces an initial training phase, which is designed to develop these receptive fields. Later stages also have some sort continual learning and development.

An interesting relationship between all the receptors in the different layers of the visual cortex is they seem to exhibit a property called statistical independence. Several authors align with the fact that independent components are indeed a basis of receptive fields which at least make up the early visual system [1] [6] [7]. This, in a very simplistic sense, says that each receptor is good for only one specific task. The actual implementation of the training system will be discussed later in section **4.1.3.**

2.2 Independent Component Analysis

2.2.1 An Intuitive Explanation of Independent Component Analysis

Independent Component Analysis (ICA) is a statistical method, which upon the collection of a mass of data, is able to extract the underlying data which is in essence "independent" from each other. The classical example of an application of ICA, is the Blind Source Separation Problem, also known as the Cocktail Party Problem. To reiterate this problem:

> Imagine a setting, which consists of some number N people speaking simultaneously about completely different subject matter. In the area of these N people, there are also N microphones, which are recording the mixtures of the speakers. You are to recover the original content of each speaker, using the N mixtures recorded by each microphone with as little crossover as possible.

This is a trivial task for most any person to do. Most people can filter out other noise (speaking other than the subject matter is still noise) and follow the conversation of a single person in a given setting. Other examples would be ignoring music while talking to someone, or vice versa. By taking the three mixtures produced by the three microphones, and allowing ICA to process them, it is possible to extract the original signals. Source extraction is just one of the aspects of ICA which make it seem attractive to use for biologically motivated systems.

Figure 3 is a very simple example of the inputs signals, the mixture signals, and the recovered signals of an implementation of the ICA algorithm, called fastICA.

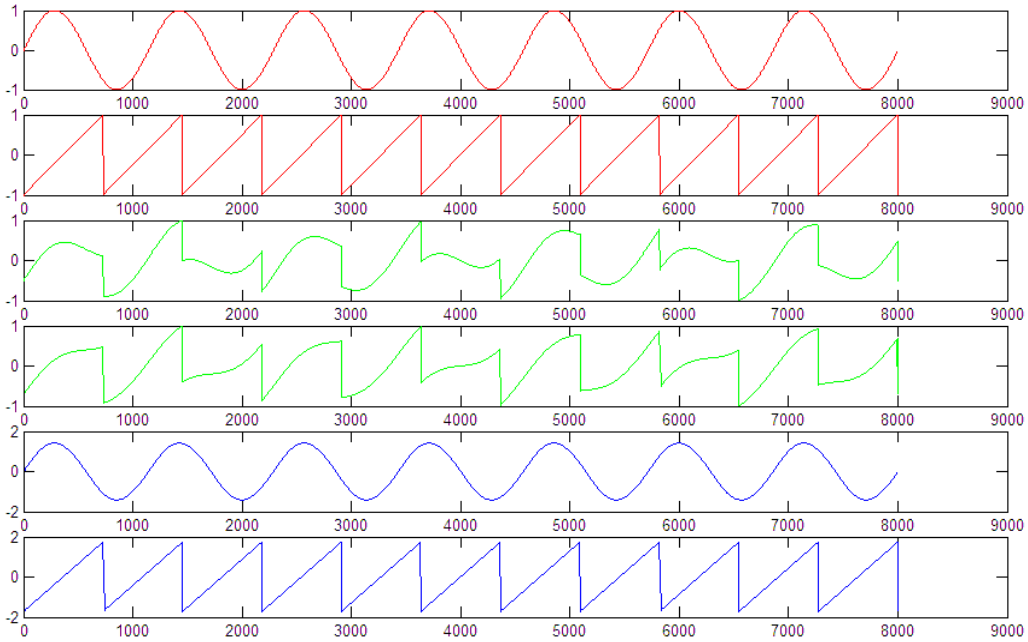


Figure 3: Source Signals **{sin(2*pi*7*[0:0.000125:1]); & sawtooth(2*pi*11*[0:0.000125:1]); }** *(Top); Mixtures of Source Signals (Middle) {* $A = \begin{bmatrix} 0.5 & 0.5 \\ 0.7 & 0.3 \end{bmatrix}$ *} where A is the mixing matrix ; Output of Signal Extraction(Bottom)*

As previously described, ICA does offer an attractive means to help in implementing a biological based system, but there are some inherent shortcomings of ICA. One of the most notable is that the extracted signals are not of the original amplitude as the input source signals. This is evident in *Figure 3*, where the two input signals were scaled between -1 and 1, whereas the output signals are scaled between -1.98 and 1.98. If the scale were to always be the same factor, that would also be acceptable, unfortunately the scale between signals may vary also (ie, for another set of data the two input signals may be scaled between -1 and 1, but the output extraction for the first signal may end up scaled between -0.5 and 0.5 and the second between -1.8 and 1.8).

Another shortcoming is the order of the extracted components. There is no guarantee that the first signal in will be the first signal extracted, the second signal will be the second and so on.

2.2.2 A Theoretical Explanation of Independent Component Analysis (Mixing /Un-Mixing) [17]

The underlying mathematics which is the basis of ICA, is deeply rooted in higher level statistics and matrix mathematics. To start at a practical level, think intuitively about the example presented above, where there are two source signals. These two source signals are mixed together based on a device called a mixing matrix.

$$s_1 = (s_1^1, s_1^2, s_1^3, ..., s_1^N) \qquad s^3 = \begin{bmatrix} s_1^3 \\ s_2^3 \end{bmatrix}^T = (s_1^3, s_2^3)$$
$$s_2 = (s_2^1, s_2^2, s_2^3, ..., s_2^N) \qquad \mathbf{s} = \begin{bmatrix} s_1 \\ s_2 \end{bmatrix}$$

Equation Set 1: Clarifying Notation of Source Signals. The s with subscript represents the signal, s_1 is signal one. The superscript represents the sample, s_1^4 is the fourth sample of signal one.

The obvious first step is to create the signal mixtures. In this text, the variable x will represent the mixtures.

$$x_1^t = a \cdot s_1^t + b \cdot s_2^t = a \cdot (s_1^1, s_1^2, ..., s_1^N) + b \cdot (s_2^1, s_2^2, s_2^3, ..., s_2^N) = (x_1^1, x_1^2, ..., x_1^N)$$
$$x_2^t = c \cdot s_1^t + d \cdot s_2^t = c \cdot (s_1^1, s_1^2, ..., s_1^N) + d \cdot (s_2^1, s_2^2, s_2^3, ..., s_2^N) = (x_2^1, x_2^2, ..., x_2^N)$$

Equation Set 2: a, b, c, and d are mixing coefficients.

This implies that the four mixing coefficients transform one vector variable s to another vector variable x. Since this is at a point-by-point basis, one index of vector s corresponds to an index of the vector x through some transform. $(s_1^t, s_2^t)^{\mathbf{T}} \rightarrow (x_1^t, x_2^t)^{\mathbf{T}}$ or more simply, $s^t \rightarrow x^t$.

$$x_1 = a \cdot s_1 + b \cdot s_2 \qquad x_2 = c \cdot s_1 + d \cdot s_2$$
$$x_1 = (a,b) \cdot (s_1, s_2)^{\mathbf{T}} \qquad x_2 = (c,d) \cdot (s_1, s_2)^{\mathbf{T}}$$
$$x_1 = v_1^{\mathbf{T}} \cdot \mathbf{s} \qquad x_2 = v_2^{\mathbf{T}} \cdot \mathbf{s}$$

Equation Set 3: This Set of Equations Represents How the Two Mixtures Are Created via Matrix Manipulation. The variable $v_1 = (a,b)^{\mathbf{T}}$ *and* $v_2 = (c,d)^{\mathbf{T}}$.

The two variables *v1* and *v2* can be combined into a simple matrix and rewritten simply as: $A = (v_1, v_2)^{\mathbf{T}}$. Finally one can produce the following Equation Set.

$$\begin{bmatrix} x_1^1 & x_1^2 & x_1^3 & \dots & x_1^N \\ x_2^1 & x_2^2 & x_2^3 & \dots & x_2^N \end{bmatrix} = \begin{bmatrix} a & b \\ c & d \end{bmatrix} \cdot \begin{bmatrix} s_1^1 & s_1^2 & s_1^3 & \dots & s_1^N \\ s_2^1 & s_2^2 & s_2^3 & \dots & s_2^N \end{bmatrix}$$
$$\mathbf{x} = (v_1, v_2)^{\mathbf{T}} \cdot (s_1, s_2)$$
$$\mathbf{x} = A\mathbf{s}$$

Equation Set 4: The Importance of This Equation Set Is the Last Line, Which Represents How Mixing Matrix **A** *Can Be Used To Create the Mixture Matrix* **x**. *To Clarify, Each Row of the Result Represents a Mixture.*

Equations Sets 2 – 4 represent how to arrive at the mixtures, which are said to exist among data, and are directly correlated to the Cocktail Party Problem with only two speakers (ie: only two sources). This is corollary to how the signals are combined. For the combination of signals, there is obviously an inverse operation for the separation of those source signals. The mathematics, which runs between the combination and separation of signals, runs parallel to one another where α is similar to *a*, β similar to *b,* χ similar to *c*, and δ similar to *d*.

$$s_1 = \alpha \cdot x_1 + \beta \cdot x_2 \qquad s_2 = \chi \cdot x_1 + \delta \cdot x_2$$
$$s_1 = (\alpha, \beta) \cdot (x_1, x_2)^{\mathbf{T}} \qquad s_2 = (\chi, \delta) \cdot (x_1, x_2)^{\mathbf{T}}$$
$$s_1 = \omega_1^{\mathbf{T}} \cdot \mathbf{x} \qquad s_2 = \omega_2^{\mathbf{T}} \cdot \mathbf{x}$$

Equation Set 5: This Set of Equations Represents How the Two Signals Are Recovered via Matrix Manipulation. The variable $\omega_1 = (\alpha, \beta)^{\mathbf{T}}$ *and* $\omega_2 = (\chi, \delta)^{\mathbf{T}}$.

The two variables ω_1 and ω_2 can be combined into a simple matrix and rewritten simply as: $W = (\omega_1, \omega_2)^{\mathbf{T}}$. Finally one can produce the following Equation Set.

$$\begin{bmatrix} s_1^1 & s_1^2 & s_1^3 & \dots & s_1^N \\ s_2^1 & s_2^2 & s_2^3 & \dots & s_2^N \end{bmatrix} = \begin{bmatrix} \alpha & \beta \\ \chi & \delta \end{bmatrix} \cdot \begin{bmatrix} x_1^1 & x_1^2 & x_1^3 & \dots & x_1^N \\ x_2^1 & x_2^2 & x_2^3 & \dots & x_2^N \end{bmatrix}$$

$$\mathbf{s} = (\omega_1, \omega_2)^{\mathbf{T}} \cdot (x_1, x_2)$$

$$\mathbf{s} = W\mathbf{x}$$

Equation Set 6: The Importance of This Equation Set Is the Last Line, Which Represents How Un-Mixing Matrix $\boldsymbol{W}$ *Can Be Used To Recover the Signal Matrix* $\mathbf{s}$*. To Clarify, Each Row of the Result Represents A Single Signal.*

Now that the mechanics of combining and extracting signals has been reviewed, the more interesting question involves two of matrices above. $\boldsymbol{A}$ and $\boldsymbol{W}$, especially $\boldsymbol{W}$ are the two important matrices. Some versions of ICA use an iterative process to determine the value of the $\boldsymbol{W}$ matrix.

In concise terms, $\boldsymbol{W}$ can extract multiple source signals in the mixture matrix $\mathbf{x}$ because each row in $\boldsymbol{W}$ is a vector which is orthogonal to one of the transformed signals axes, s_1' or s_2'. The variables s_1' and s_2' are $\begin{bmatrix} a \\ c \end{bmatrix}$ and $\begin{bmatrix} b \\ d \end{bmatrix}$ respectively. This is derived from the fact that the variables are in line with their respective axes, such that s_1 represents a horizontal axis, and that s_2 represents a vertical axis orthogonal to s_1. A vector coordinate which lies in the s_1 axis, simply can be defined as (1, 0). The opposite is also true where, (0, 1) lies in the s_2 axis. To better clarify this:

$$s_1^{'} = \mathbf{A}s_2 \qquad s_2^{'} = \mathbf{A}s_1$$

$$s_1^{'} = \begin{bmatrix} a & b \\ c & d \end{bmatrix} \cdot \begin{bmatrix} 1 \\ 0 \end{bmatrix} \qquad s_2^{'} = \begin{bmatrix} a & b \\ c & d \end{bmatrix} \cdot \begin{bmatrix} 0 \\ 1 \end{bmatrix}$$

$$s_1^{'} = \begin{bmatrix} 1a + 0b \\ 1c + 0d \end{bmatrix} \qquad s_2^{'} = \begin{bmatrix} a + 1b \\ 0c + 1d \end{bmatrix}$$

$$s_1^{'} = \begin{bmatrix} a \\ c \end{bmatrix} \qquad s_2^{'} = \begin{bmatrix} b \\ d \end{bmatrix}$$

$$\mathbf{A} = (s_1^{'}, s_2^{'})$$

$$\therefore$$

$$\mathbf{A} = \begin{bmatrix} a & b \\ c & d \end{bmatrix}$$

Equation Set 7: Deriving Variables Which Are Incident Along Specified, Orthogonal Axes.

"Orthogonality", is one of the key principles to extracting multiple signals from one another. This is simply because of the property that, once the inner product of two vectors which are orthogonal to one another are computed, the output is simply zero. Let $\omega_1^{\mathbf{T}} = (\alpha, \beta)$, by taking the inner product of $\omega_1^{\mathbf{T}}$, and say $s_2^{'}$ one would get 0. Since this is true, what about the inner product of $\omega_1^{\mathbf{T}}$, and say $s_1^{'}$? Since they are not orthogonal to each other, they will yield some constant *k*.

$$\omega_1^{\mathbf{T}} \cdot s_1^{'} = k$$

$$k = \left|s_1^{'}\right| \cdot \left|\omega_1\right| \cdot \cos\Theta$$

Equation Set 8: Deriving k, the Scaling Factor for the Extracted Signal.

Finally what falls out, is a scaled version of source signal s_1 by a factor of *k*. The signal s_1 is extracted from the mixed signals in matrix **x** just by taking the inner product, column-element wise of **x** with a vector $\omega_1^{\mathbf{T}}$ that is orthogonal to $s_2^{'}$ in **x**. Since it is now known that the actual signal which is extracted is a scaled version of the original, signal s_1 is really extracted as signal $s_1 \cdot k$.

$$s_1 = \left|s_1^{'}\right| \cdot \left|\omega_1\right| \cdot \cos\Theta \cdot \mathbf{w}_1^{\mathbf{T}} \cdot \mathbf{x}$$
$$s_1 = k \cdot \omega_1^{\mathbf{T}} \cdot \mathbf{x}$$

Assume k = 1 for simplicity: $s_1 = \omega_1^{\mathbf{T}} \cdot \mathbf{x}$

Similarly: $s_2 = \omega_2^{\mathbf{T}} \cdot \mathbf{x}$

Given: $W = (\omega_1, \omega_2)^T$, $\mathbf{s} = (s_1, s_2)^T$

Therefore: $\mathbf{s} = W\mathbf{x}$

Equation Set 9: Explaining Final Steps of Extraction with Two Signals.

This is one of the underlying principles as to how independent component analysis works. By determining which signal is orthogonal to which, will generally allow for the extraction of multiple signals in a given number of mixtures. More detailed information can be obtained in [17].

2.2.3 How Independent Components Apply To Images

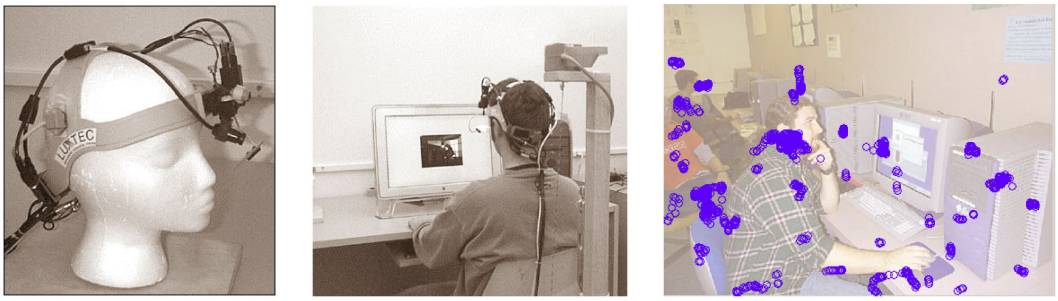
Patches of images represent the observed mixtures of signals, which are denoted as the term **x**. By passing many observed mixtures, **x**, to ICA, a set of *M* source signals are extracted, where *M* is the number of components that ICA will extract. The components extracted are parallel to the fields found in the mammalian visual cortex. This is to say that the receptive fields discussed in 2.1 are the devices which when constructively combined, allow one to perceive the visual representation of the surroundings. To state this more concisely, for some image *I*, $I(x,y) = \sum_i a_i C_i(x,y)$, where a_i is a specific amplitude of independent component C_i [19]. One can draw how this parallels with the Fourier Summation example drawn early in 2.1.1, where some arbitrary function *x* can be realized by: $x(t) = A_0 + \sum_{k=1}^{\infty} A_k \cdot \cos(k\omega_0 t + \phi_k)$.

By taking only a certain number of these receptive fields (as opposed to all of them), and constructively combining them, with their respective response to an image will yield a reconstruction of the original image. One can see how this parallels the first part of 2.1.1, which describes the Fourier Summations of multiple sinusoidal waveforms.

Finally by training on real data and applying ICA to the observed data **x**, Gabor-like receptive fields emerge [19]. The materialization of these Gabor-like receptive fields makes for a promising set of results, because in other experiments, artificially constructed Gabor receptive fields, and Difference of Gaussian (DoG) receptive fields have given positive results. Using the model presented by ICA, the Gabor-like receptive fields appear as the in-mixing matrix, $\boldsymbol{W}$[11].

III Previous Work

Saliency is a pronounced feature or part similar to a highlight. There have been many different implementations of systems where the primary goal is to detect saliency within still images. These systems relate to where a person's gaze is directed. Most of these systems can be compared to eye tracking and gazing experiments. Researchers will generally display a blank screen for some amount of time then expose the subject to some visual input. All the while the eye tracker, and possibly a head tracker, is recording information about where the subject looks within the image. Of course other information may be recorded along with the path the eye moves to go from one point to another.



(a) *(b)* *(c)*

Figure 4: (a)ASL Model 501 Eye Tracker and Polhemus 3-Space Magnetic Head Tracker; (b)Complete Test Fixture; (c)Output From Test Figures From [12]

The objective of these eye tracking experiments is to determine what is the most interesting regions of an image. Usually in the mammalian visual system, once an area has been found to be interesting the eye will move such that the are of the fovea is focused on the point of interest. The next level of indirection is to use this information and have a computer mimic the results from *Figure 4c*. Earlier works by [9] have successfully created saliency maps which are able to loosely mimic the

results shown in *Figure 4c*. The remainder of this section will present the bottom-up portion of two previously implemented systems as well as some extensions of this work.

3.1 Models of Bottom-Up and Top-Down Visual Attention

Research and work recently performed by Laurent Itti of the California Institute of Technology has implemented a system of detecting regions of interest. "The bottom up system which was implemented is a preattentive selection mechanism based on the architecture of the primate visual system" [8].

3.1.1 Generating the Bottom-Up Saliency Maps

Itti, like most researchers in this field find that extracting low level features is one of the necessary steps in building up the salient feature maps. One of the key points of interest is how far basic model dates back. The original schematic representation was proposed by Koch and Ullman as far back as 1985 [9]. Itti had elaborated on the original schematic and settled on the following schematic.

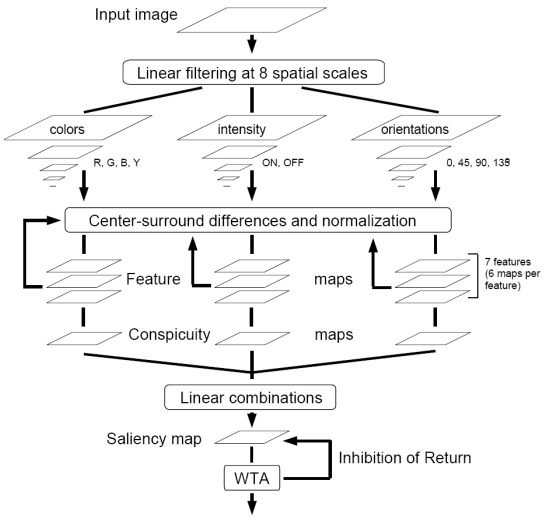


Figure 5: Schematic Diagram of Itti's System for Bottom-Up Saliency Detection figure from [8]

One will notice the multiple planes of descending sizes for the information with respect to “colors”, “intensity”, and “orientations”. This is basically a method for detecting the scale of the salient features in the presented image. In this particular work, there were a depth of 9 different scales progressing from a scale of 1:1 (original image size) to a scale of 1:256, consecutively in steps of powers of two (ie: $[1,2,...,2^n]\big|_{n=[1,8]}$). This is performed at a level for each of the kernels which each component of the image had been filtered with.

Itti uses multiple facets of the image to generate the response maps. These components are intensity, color, and edge orientation. This system’s breadth was to be used with RGB images from multiple sources, so the intensity values generated were not geared toward one specific input source. Itti simply calculates the image intensity I as the average of the r, g, and b values for each respective pixel within the image [8]. This unfortunately does not correspond to the response of the human eye, which peaks at approximately 550nm (yellow-green light). Under the suggestions of Itti, it may be beneficial to better model this when trying to model the intensity selective neurons [8]. Finally, the output of the Intensity is scaled from it’s original size in the fashion previously discussed.

Using a biological parallel to color difference channels, which respond highly to differences between two specific colors, relies on extracting specific color data from the input image. After normalizing values for the r, g, and b channels by I (previously calculated). This decouples the hue from the intensity. Following that step, values less than 1/10 of the maximum of the entire image are simply set to zero. At this point the four broad color channels are created by the following:

$$R = r - \frac{(g+b)}{2}$$
$$G = g - \frac{(r+b)}{2}$$
$$B = b - \frac{(r+b)}{2}$$
$$Y = r + g - 2 \cdot (|r-g| + b)$$

Equation Set 10: Equations to Extract Information to Generate Color Difference Channels Filtered Output

Generating this representation of R G and B channels is very similar to the case previously presented with respect to the intensity channel. The color-sensitive photoreceptors, or cones, in the retina do not have uniform sensitivity across the spectrum of colors. Again, just as with the intensity maps, once the response to the kernels is generated they are scaled in the same iterative fashion.

In this particular work, the orientation receptors have been developed in previous work by Greenspan *et al* [5]. The orientations of different features in an image are determined through the intensity map I previously generated. Again this will be scaled across multiple resolutions. The orientations chosen are quite general, at 0°, 45°, 90°, and 135°. These particular filters are Gabor filters, which seem to approximate the neuronal structure in the primary visual cortex [10].

Figure 6: Some Oriented Filters [figure adapted from 8]

One of the unique features of Itti's work is the implementation of center-surround receptors. The modeling in simplest terms takes the difference between a fine scale and course scale response for a given feature. The intensity channels use

the opponency maps simply with the course (c) and fine grain (*s*) difference. There is the same sort of notion which deals with the orientation based information. The only difference is the output from each of the orientation is only combined with that of the same orientation. The most interesting part of this work is the workings of the color difference channels. The opponency maps are between the red and green and separately the blue and yellow color channels. These four different center-surround receptors can be represented by the following equation set:

$$
\begin{aligned}
Iout(c,s) &= \left|I(c)\Theta I(s)\right| \\
Oout(c,s) &= \left|O(c,\theta)\Theta O(s,\theta)\right| \\
RGout(c,s) &= \left|(R(c)-G(c))\Theta(G(s)-R(s))\right| \\
BYout(c,s) &= \left|(B(c)-Y(c))\Theta(Y(s)-B(s))\right|
\end{aligned}
$$

Equation Set 11: Four Equations Which Govern the Output of the Center Surround Difference Channels

The numerous maps generated by these different operations at all scales (and in some cases orientations), need to be finally combined into a single saliency map. Each of them alone serves as a map, which can itself identify possibly salient features in an image. The point of the entire system is to create a bottom-up system which can emulate visual search and gaze, similar to how a human would inspect an image. There are many ways which this combination could be accomplished. Itti suggests four different methods to perform this combination. The simplest approach to this combination is to normalize all of the maps (possibly to values between 0 and 1) and naïvely sum them all together. The **Naïve Summation** approach, while simple, has faults in that it is susceptible to noisy data. **Learning Liner Combinations** involves supervised learning, when the targets which are if interest in an image are known and present. This in some respects is an extension of the **Naïve Summation** approach. Each set of maps which make up the final map is multiplied by a specific weight. The

weight is determined from a number of example images in which the desired target has been outlined [8]. Training basically is the process of giving the maps of highest response a greater weighting, while suppressing the weights of maps of lower response. This type of combination works well for a system which is able to have some sort of top-down supervision or human interaction. A third approach to this combination of maps problem is called **Contents-Based Global Non-Linear Amplification**. It provides a solution to scenarios where no top-down supervision is available [8]. This approach promotes features in a map that are strong while suppressing common peaks detected throughout the image. The algorithm essentially consists of finding a global maximum for an image map, as well as the image map's mean value and multiplying the squared difference of the two terms. The final method described as **Iterative Localized Interactions** is conceivably very simple, but computationally intensive. The method take a Difference of Gaussian filter and linearly filters each of the maps generated by the early system input, then filters that output again, and again, iteratively ten times.

There are multiple benefits of using this particular algorithm for this type of determination. The largest factor to using **Iterative Localized Interactions** over **Contents-Based Global Non-Linear Amplification** is that it has some biological motivation. **Contents-Based Global Non-Linear Amplification** uses the concept of finding global maximums. Since cortical neurons are connected in local bundles, it is not very conceivable that they, themselves, could determine global maxima. This lands strikes for the two previously mentioned methods.

3.1.2 Some Results

While there were many results from this particular work, which dealt with the interaction between the top-down saliency with some semantic reasoning and the bottom-up approach briefly described, this section will only display and narrate the results of the bottom-up saliency map construction.

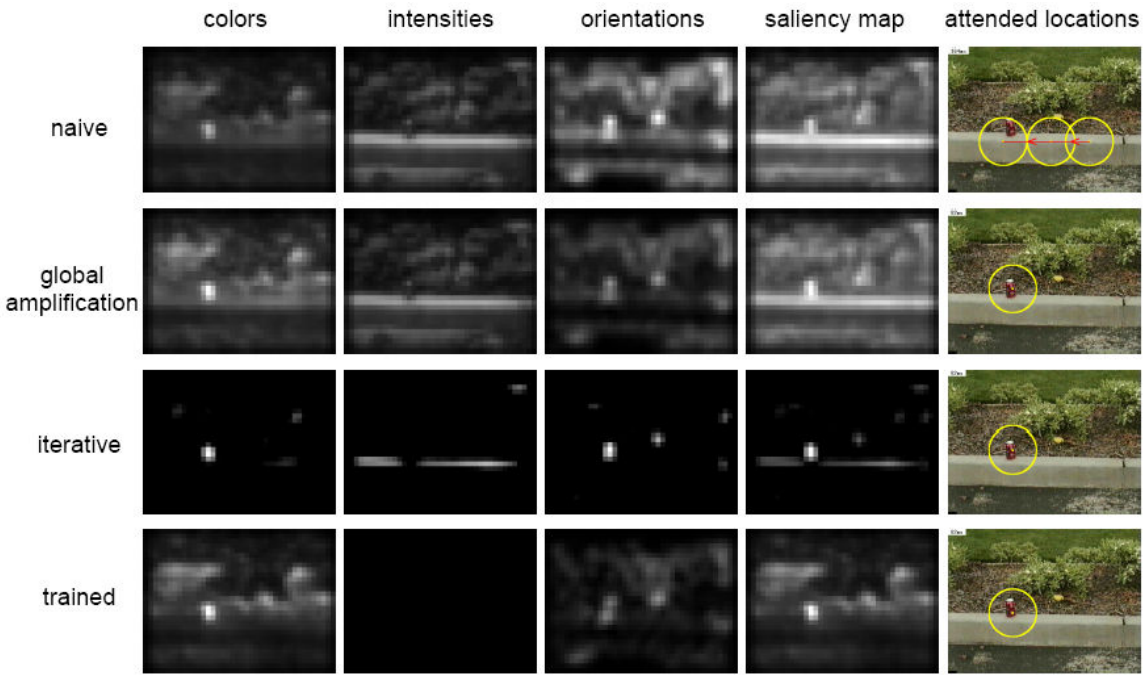


Figure 7: Comparing Methods of Map Summations and Intermediate Steps figure from [8]

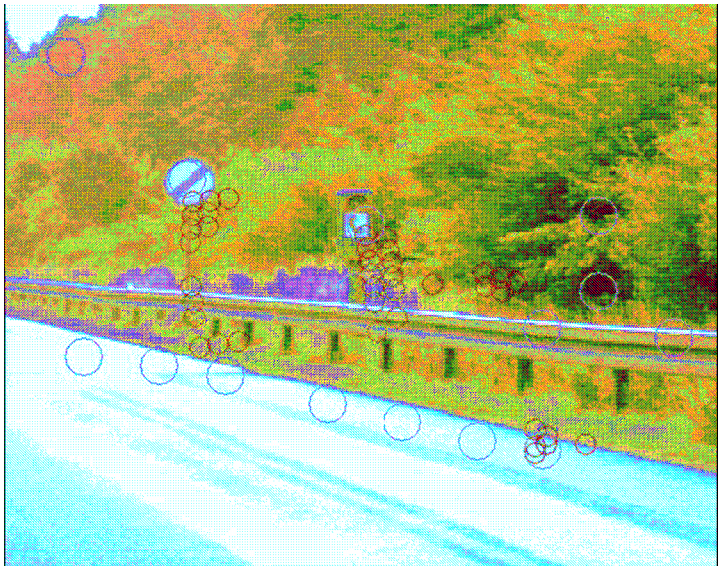


Figure 8: Comparing Pupil Tracking Results With Bottom-Up Saliency Generation. The Smaller Red Circles Indicate Pupil's Direction of Gaze & Larger Blue Circles Are Generated From the Actual System figure from [8]

3.2 VENUS: A System for Novelty Detection In Video Streams with Learning

VENUS is a system designed to detect novelty in video. It has the ability to "remember" novel events, and record them over time. As time goes on, there is a decay of how strong the remembrance of the event is. This system has some very strong ties to a subset of the system which, Itti had implemented [8]. Beyond the early visual stage, VENUS diverges from the path, which Itti had taken, and moved on to process motion data as well, then combines the still saliency information with that of the motion data. By viewing some videos processed by Itti's system, one will find that many of the circles which represent areas of saliency move around frame to frame, which makes sense. It seems that the algorithm was simply applied to each frame without taking motion into context.

3.2.1 Still Processing Channel

As stated before, VENUS takes its still processing pathway from Itti's work. The minor difference is that for the saliency map generation VENUS uses a variant of **Contents-Based Global Non-Linear Amplification** in conjunction with the **Iterative Localized Interactions** method. After the **Contents-Based Global Non-Linear Amplification** is applied to the given saliency map for a particular feature channel, the output is run through a set number of iterations in the **Iterative Localized Interactions** method. At this point, VENUS simply processes each frame of video through this bottom-up saliency map generation algorithm, without any context of motion. In a biological sense this parallels the ventral pathway, which deals primarily with object recognition in the visual pathway.

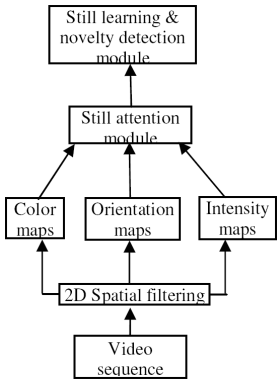


Figure 9: Still Processing Schematic for VENUS figure adapted from [8]

After the output saliency map is computed, it is fed into a still learning and novelty detection module which is actually quite simple. It takes in to account all the previous history which has been learned and creates a still habituation map. This still habituation map is simply a running average of all the input received up to, but not including the current still saliency map [4]. Depending upon the learning rate (α) chosen, this will determine how important the history is versus how important the

current frame being processed is. Since *f(α)* is governed implicitly as a function of time, they are dependent on the frame rate of video. For NTSC frame rates (29.97 FPS), reasonable values for α to create a learned map would be on the order of 0.98 for the weight of the history, and the current frame would be only 0.02. The more generalized form of this would simply be:

$$newAvg(\alpha) = \alpha \cdot prevFrames + (1-\alpha) \cdot currFrame$$

Equation Set 12: Learning Rate Implementation. Remember, the newAvg will be fed back in as prevFrames

3.2.2 Motion Processing Channel

The responsibility of the motion processing channel is to determine the novelty of a specific event occurring. In the implementation of VENUS, if the same feature value is observed repeatedly over time, the system will "get used to" (habituate) this particular event and eventually stop interpreting it as a novel event [4]. One of the interesting facets of the motion processing channel is its ability to forget. Although one usually would not put forgetting into the general equation of a successful system, it is part of the biological model. Basically this forgetting term parallels the habituation term, and can even be thought of as dis-habituation.

VENUS breaks up each of the images into 8x8 sectioned tiles. Upon some sort of motion occurring in these tiles a Gaussian corresponding to that particular tile will be updated. The update will be sent to either a North, South, East, or West Gaussian depending on the direction of motion in the tile. The information used to model in these Gaussians is the velocity of the object moving through a specified tile. Based on multiple similar objects passing through the same region, and moving in the same approximate manner, will allow the distribution(s) to gain solid shapes by incorporating more statistical information. If a similar object moves in a similar

manner, but at a speed 4 times greater than that of the previously moving objects, this will grow a new Gaussian which will represent a novel event. Now, there are two Gaussians which are required to represent that specific 8x8 tile. The similarity of these two events is solely based on values of the mean and standard deviation for these particular tiles. The only bounds of events able to be added to the pool is based on the physical limitations of the hardware and in the implementation.

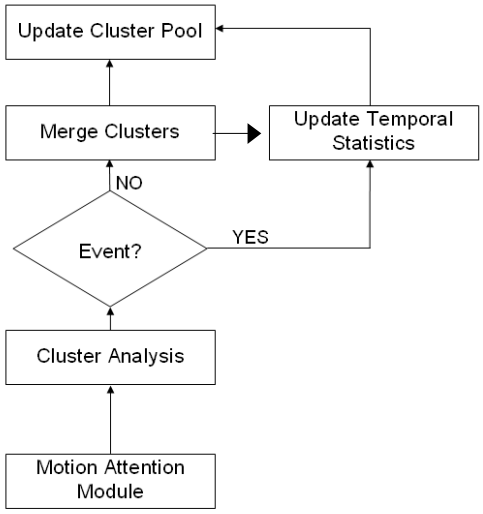


Figure 10: Motion Learning and Novelty Detection Module figure from [4]

Finally in this particular implementation, the forgetting factor comes into play. This is represented by a sigmoid-like habituation curve. This curve is also associated to each of the Gaussians constructed for the 8x8 tile region.

$$H(t) = 1 - \left[\frac{1}{1 + e^{-a \cdot t}} \right]$$

$$a_t = 1 - \left[\frac{e}{f} \right]$$

Equation Set 13: Habituation Function and Decay Function where e is the number of times the cluster was merged and f is the number of frames since the cluster's creation [4].

By applying this algorithm to the input video, over time certain events will become habituated and essentially learned away.

3.2.3 Some Results

This section will have some of the results from VENUS. Since the output of VENUS is directly related to the system which is being implemented in the current work, the final output will be displayed.

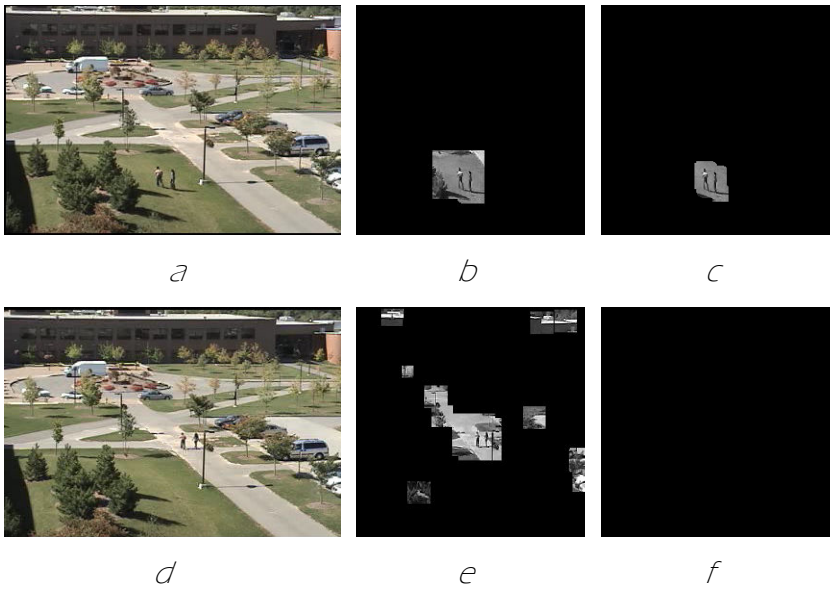


Figure 11: A set of output from VENUS. The leftmost images are the input video. The middle images are the outputs of a motion segmentation algorithm. The rightmost images are the novel event maps. The top row represents frames 937 of a video. Since this is the first time people were walking on the lawn, it appears in the novelty map (c) as well as the motion segmentation map (b). The bottom row represents frame 1237, where the people have already walked out of the lawn, and new people have walked in. Again, the motion segmentation map (e) shows the people walking through the lawn, but the novelty map (f), is blank, meaning that this event has been habituated. Figures from [4].

IV System Implementation & Analysis

This system for detecting saliency and novelty is implemented in Mathwork's Matlab, primarily for the quick development time, and its ability to perform efficient matrix mathematics. Matlab also has extensible toolboxes that are both built in and provided by third parties which aided in the implementation of this work. Some of the toolboxes used were the Image Processing Tool Box, and a third-party implementation of fastICA, made public by, Hyvärinen from the University of Helsinki, Finland.

The remainder of this section will focus on the implementation of the current system for detecting saliency in video streams. The first section deals with the learning of the system's still channel. The second section details how the learning applies to still images and extend that concept to video. Finally, a description of the top-down saliency model will be discussed.

4.1 Learning Independent Receptors

Independent receptors have been discussed in past sections (**2.2.3**). The receptors are learned based on exposure to input data. Since this is primarily an imaging based system, the information is in the form of image patches.

Although the visual pathway carries incredibly large amounts of information down the optic nerve, a lot of it is filtered out due to attention and other mechanisms. These mechanisms are developed over time. As the receptors of the visual pathway start to become more defined, they develop this selectivity. In early stages, this

selectivity does not exist, therefore at an early stage of development of these receptive fields, all information is taken in and transmitted down the optic nerve. This vast array of information is what helps to develop the receptors which many researchers attempt to model [1][4][7][8][9][11][13][16][18][19].

4.1.1 Learning Independent Intensity Filters

Figure 12 shows the learning pathway for a small set of data. The tan squares indicate images where information is taken from. Data of the entire image is not recorded, but rather, small light blue squares (within the tan squares) indicate what information is used for processing. All patches taken from an image are of the same dimensions, but non-overlapping. Each of these patches is then vectorized and placed in a 2-D matrix. The dimensions of this array are simply NxM, where N is the product of the length and width of a single sample patch, and M is how many sample patches were taken from the entire set. For the initial training phase each, non-overlapping, sample patch was 9x9 pixels, and over a set of 600 high quality color JPEG images, 12,000 sample patches were extracted.

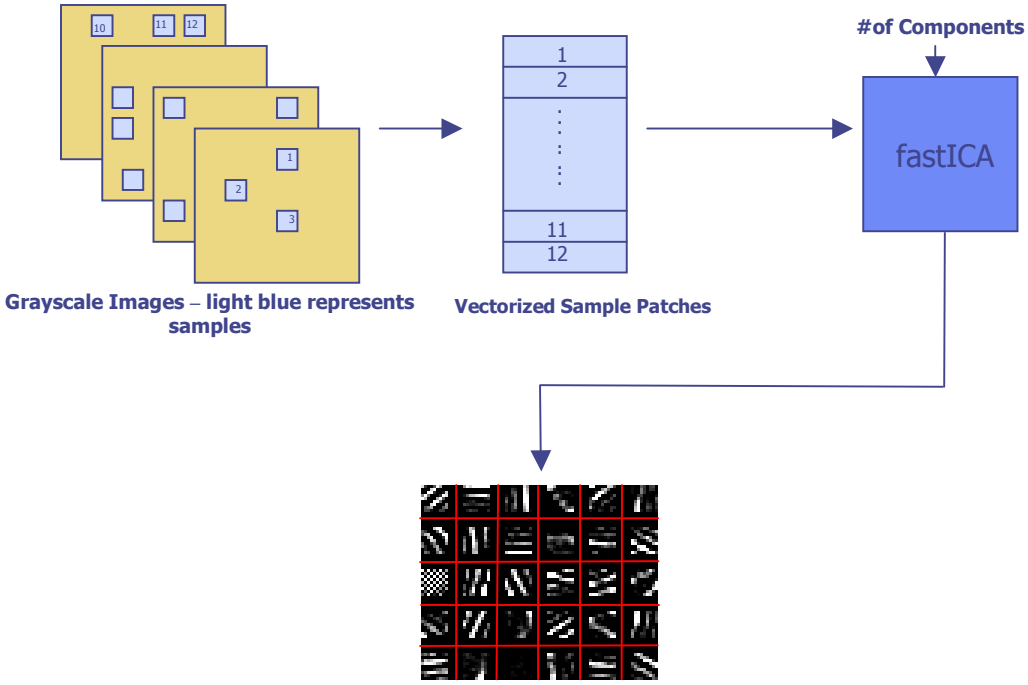


Figure 12: Pathway for learning the orientation/intensity spatial filters to be later used.

The matrix of these sample patches are then transposed and fed into the fastICA algorithm. The transposition of this matrix defines the difference of utilizing temporal ICA, versus spatial ICA. For this particular work, the observed signals to be compared exist between the different image vectors (the vectorized patches). This implies that temporal ICA is used for extracting the features. Notice that the number of components that ICA will be allowed to generate is fixed. The fastICA algorithm is actually allowed to generate as many independent components as it is able to find, by limiting the number of components generated (30 for this work), gives more interesting receptors.

The output from fastICA is a 2-D matrix of size ExF, where E is the product of the length and width of the sample patches, and F is the number of components fastICA was passed to extract. By taking this matrix and row-by-row reshaping the vectors into the same dimensions as the original sample patches, this generates the set

of filters shown at the bottom of *Figure 12*. Each red box is a separate filter. The graphic at the bottom, visualizes the. In the actual system, these filters are left in vector form, and only ever realize the 2-D filter right before applying to an image.

The filters learned through this mechanism have good foundation for detecting orientation and intensity features that compose images. Each filter presented in the *Figure 12* has very distinctive characteristics. Some of these filters respond highly to vertical edges, whereas others have greater responses to horizontal elements of an image.

4.1.2 Learning Independent Color Difference Filters

Although in many works color has been a topic of modest (at best) interest, there is a wealth of information to be gained from taking into account color information. While people who are colorblind are able to perceive their surroundings, there is some information which is skewed due to the shift of sensitivity of their L-cones and M-Cones. L-cone and M-cone are the most common types of deficiency and usually causes Red-Green Colorblindness. What is not perceived by one who is colorblind is the absence of all color (complete grayscale), monochromasy is extremely rare. The models developed in this experiment are based on a healthy visual system, and unlike previous works learn and use color receptors.

The pathway used to generate the color difference channels is very similar to that of the intensity/orientation filters. There are some minor changes in how the input data is fed into fastICA, but beyond this, it is essentially the same process. The only constraint on this particular process is that the patches used for learning these filters comes from the same patches as the patch-set used for learning the intensity/orientation filters, and the number of components to be extracted is the same.

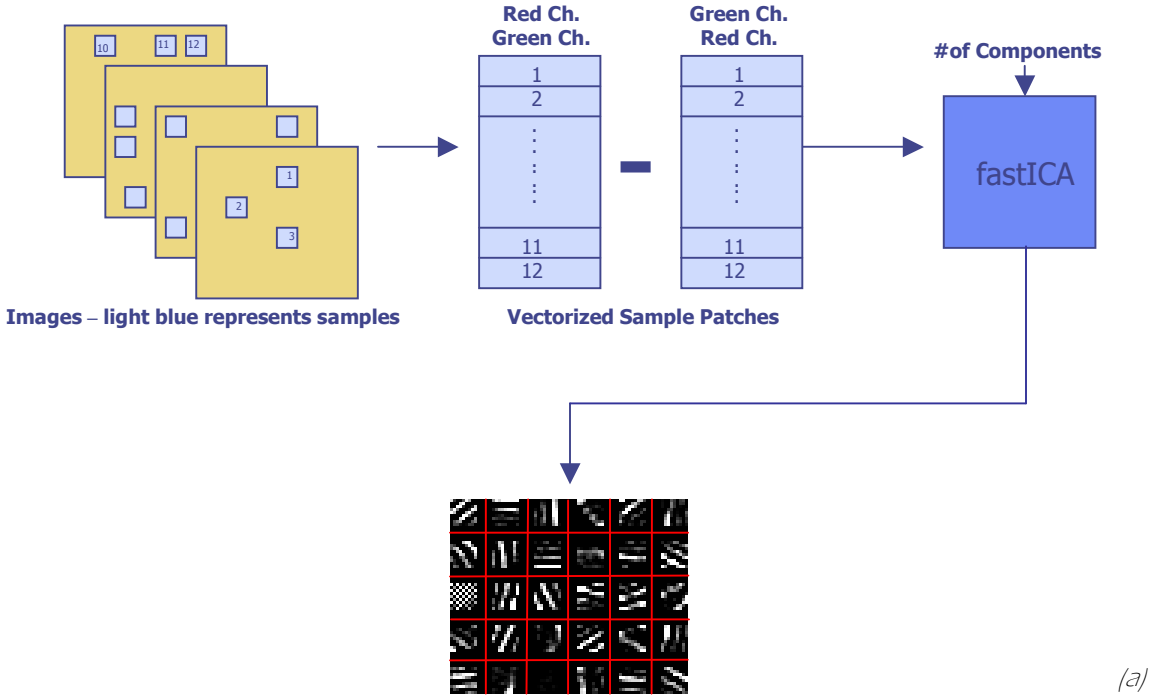


Figure 13: Pathways for learning the color difference spatial filters to be later used. (a) The Red/Green Color Difference Receptor Generation Model. (b)The Blue/Yellow Color Difference Receptor Generation Model.

4.1.3 Overview of the Filter Learning System

Since both the intensity/orientation filters and the color difference filters share much information the actual system implementation couples them tightly together.

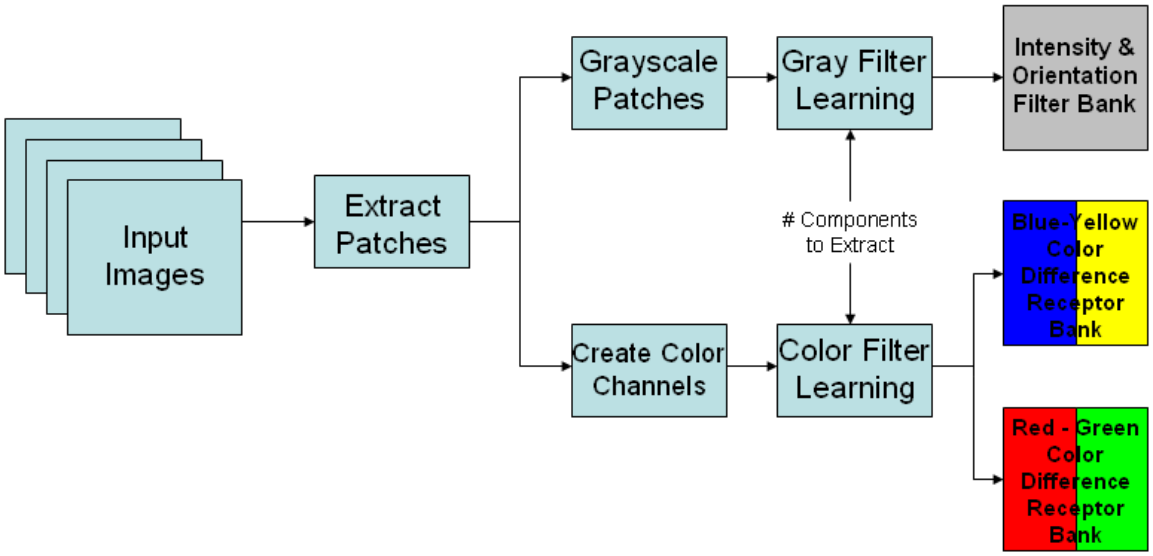


Figure 14: Schematic representation for the entire filter learning system.

Figure 14 represents how the pathway works for learning the different filter types. The patch extraction phase sends all of the patches to both pathways for further processing. Since all images processed in this system are color, the preprocessing step to the actual filter learning is a simple conversion from RGB data to Intensity data. This is accomplished using the Mathwork's `rgb2gray` function. As a preprocessing step to the Color Filter Learning processing block, it is necessary to generate proper color channels for the differencing technique. These color channels are created according to Itti [8] by *Equation Set 10*. The output of the Color Filter Learning block is two disjoint filter banks. Each filter bank represents a set of filters, which exhibit properties similar to either a Red – Green/Green – Red Color Difference Receptor or a Blue – Yellow/Yellow – Blue Difference Receptor.

The output of the filter learning system is a set of three equally sized filter banks. The Intensity & Orientation Filter Bank filters are used in later stages to extract the information their name implies. The color difference channels are used to extract information either regarding high concentrations of the colors, Red, Green, Blue, or Yellow, in a given image. The process of this extraction will be discussed in the next section (**4.2.1**).

4.2 Application of the Learned Filters

Saliency extraction is based on a bottom-up methodology using the low-level feature filters learned in section **4.1.3**. The general framework of the bottom-up implementation is similar to that implemented by Itti [8]. The output of each filter application to an image is a response map to some salient feature that the filter represents.

4.2.1 Filter Selection & The Still Saliency Pathway

The filter learning system extracts 30 source signals, therefore 30 vectors are created in the un-mixing matrix, for the different output banks shown in *Figure 14*. The number of source signals extracted can be easily changed, but 30 seems to yield fine results for the generation of basis receptive fields (described in **2.2.3**). Once the intensity and orientation filter bank have been generated, they are visualized. The visualization provides the user of the system with a general idea of what the filter might be good at detecting (horizontal edges, vertical edges, diagonals, etc.). The user then selects four filters which appear to respond best to 0°, 45°, 90°, and 135° edge orientations. In the future, filter selection should be an algorithmic approach with some form of component pruning similar to that implemented by Liu and Wang [13]. Based on empirical results, the orientation that arises in the multiple receptors of the

color difference channel does not place enough of a bias on certain colored shapes, to go through the same process of looking for specifically structured color receptors. If a shaped color receptor were to be chosen, the best candidate would be one which exhibits Gaussian-like form. In this particular implementation, both the R – G and the B – Y color difference filters are chosen at random.

Once the basis filters have been chosen to represent the multiple receptive fields, they are scaled up to multiple sizes. Each filter has three corresponding sizes, starting with the original 9x9 pixel patch originally generated by ICA, then scaling each filter up using bi-cubic interpolation to 17x17 and finally 33x33 pixel receptive fields. The relation between these filter sizes is a $2^N + 1$ relationship, and it can continually be done to create larger filter sizes. The multiple filter scales are designed to mimic the larger receptive fields found in the later stages of the visual pathway, which respond to larger and sometimes more complex features. The current implementation of the still saliency pathway uses a total of 24 filters as follows (note the grouping of the color filters):

Receptive Field Size	**Receptive Field Types**					
9 x 9	*Orientation 0°*	*Orientation 45°*	*Orientation 90°*	*Orientation 135°*	*Color B – Y/Y – B*	*Color R – G/R – G*
17 x 17	Orientation 0°	Orientation 45°	Orientation 90°	Orientation 135°	Color B – Y/*Y – B*	Color R – G/*R – G*
33 x 33	Orientation 0°	Orientation 45°	Orientation 90°	Orientation 135°	Color B – Y/*Y – B*	Color R – G/*R – G*

Figure 15: Overview of the different filters used in the implementation of the still saliency pathway. ***Italicized*** *text indicates that these particular filters were the original components obtained from ICA. All other filters are derived from the* ***Italicized*** *filters.*

Once all the filters have been chosen and scaled to their proper sizes, they are put through an iterative normalizing routine, independent of one another. The

normalization which occurs is quite simplistic, and scales each value in the receptor map by an either, decreasing or increasing variable, on every iteration that the convolution of the filter with itself does not sum to one. This is a very rudimentary normalization routine and can definitely be improved upon in the future.

Finally these filters are applied to images. The images can be from any source so long as they are RGB images that Matlab can read in. The application of the filter to the image data is done via a process called convolution. Convolution, in simplest terms, "is an integral that expresses the amount of overlap of one function g as it is shifted over another function f [21]." For this work the f and g are the values of the image and the filter. A filter that exhibits structure similar to a horizontal edge, when convolved over a section of an image with horizontal edges, will produce a high value of overlap as compared to a region where there are no edges present. The value of this overlap is what determines the interesting or salient features of an image with respect to the filter it is convolved with.

The idea of having these multiple filters is to find interesting features of an image and combine them all together to create a single saliency map. The steps for this are quite simple and may possibly be categorized as naïve, but the results of this pathway seem promising. To process a still image, the image is fed through the system and is convolved with each of the 24 filters listed in *Figure 15*. Prior to convolving the input image with each of the filters the image must go through two preprocessing stages. One of the preprocessing stages generates an intensity image, and the other preprocessing stage generates a set of color channels as governed by *Equation Set 10* [8].

After all of the maps are generated they must be merged together to create a final saliency map. Itti proposes several methods to merge the saliency maps together [8], and Vaingankar [18] actually combines two of Itti's methods for merging the

saliency maps together. In this implementation a simplistic maximum normalization method is employed. What this means is rather than normalizing values between 0 and 1, maximum values are found, which set the scale and the maps are normalized between 0 and the maximum value of the set of feature maps. Take for example just two maps, one the convolution of an image and a 0° orientation filter and the other the convolution of the same image and a 135° orientation filter. The 0° response map has a range [0 180], whereas the 135° response map has a range [0 210]. Since the 135° response map has the higher maximum value, the values in the 0° response map will be scaled between 0 and 210 (the maximum of the 130° response map). There are a number of drawbacks to using this particular normalization routine. One is that it does not seem biologically plausible due to the fact that it employs finding global maxima. Since the cortical neurons are only locally connected in the early visual system, this model does not seem very accurate [8]. This particular implementation employs this method for simplicity, speed, and hardware memory efficiency. Once all these values have been scaled, they are summed together to create a full color saliency and a full orientation saliency map. These two maps go through the same normalization process discussed above and are again summed together to realize the Complete Saliency Map (at the bottom of *Figure 16*).

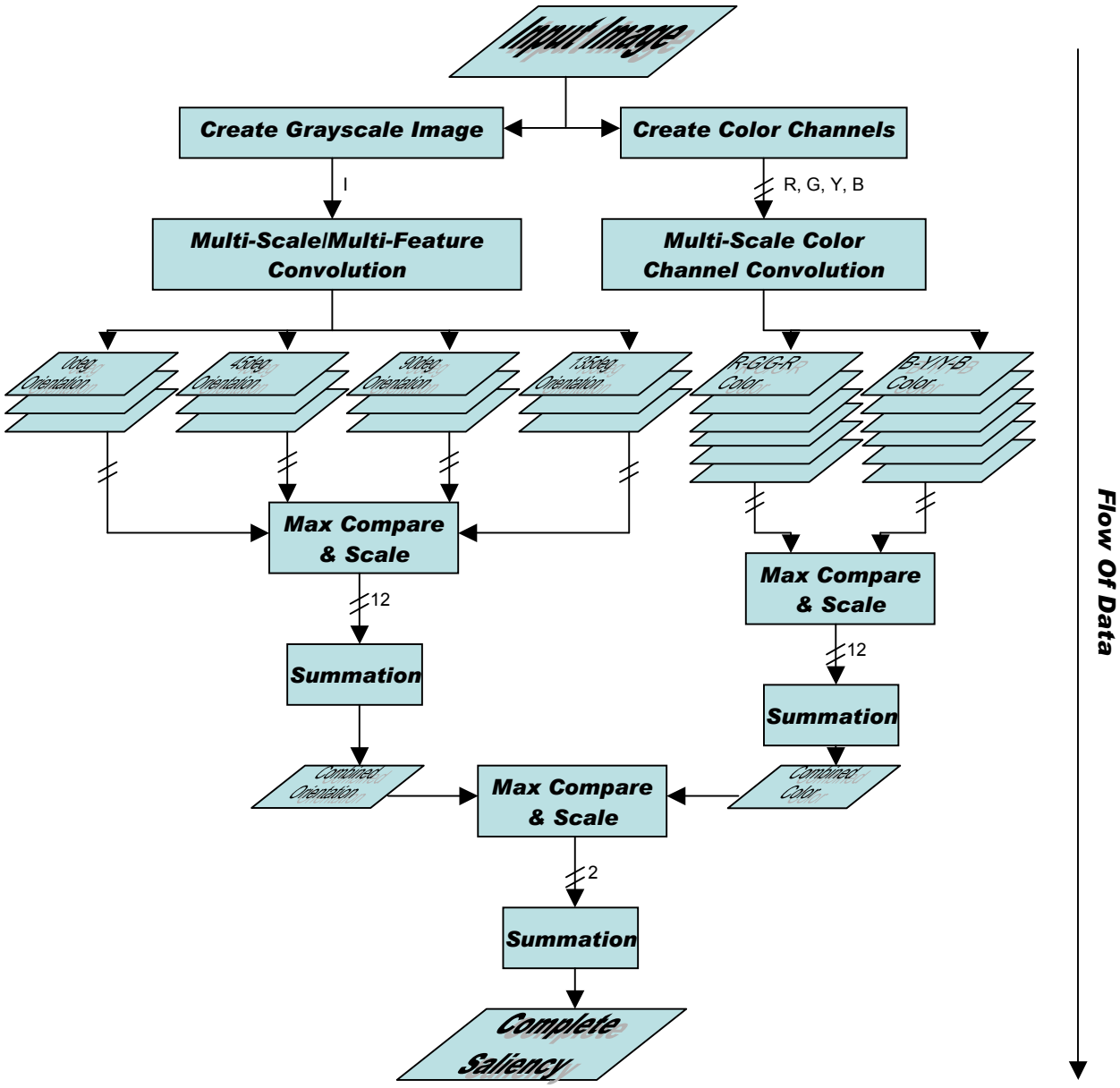


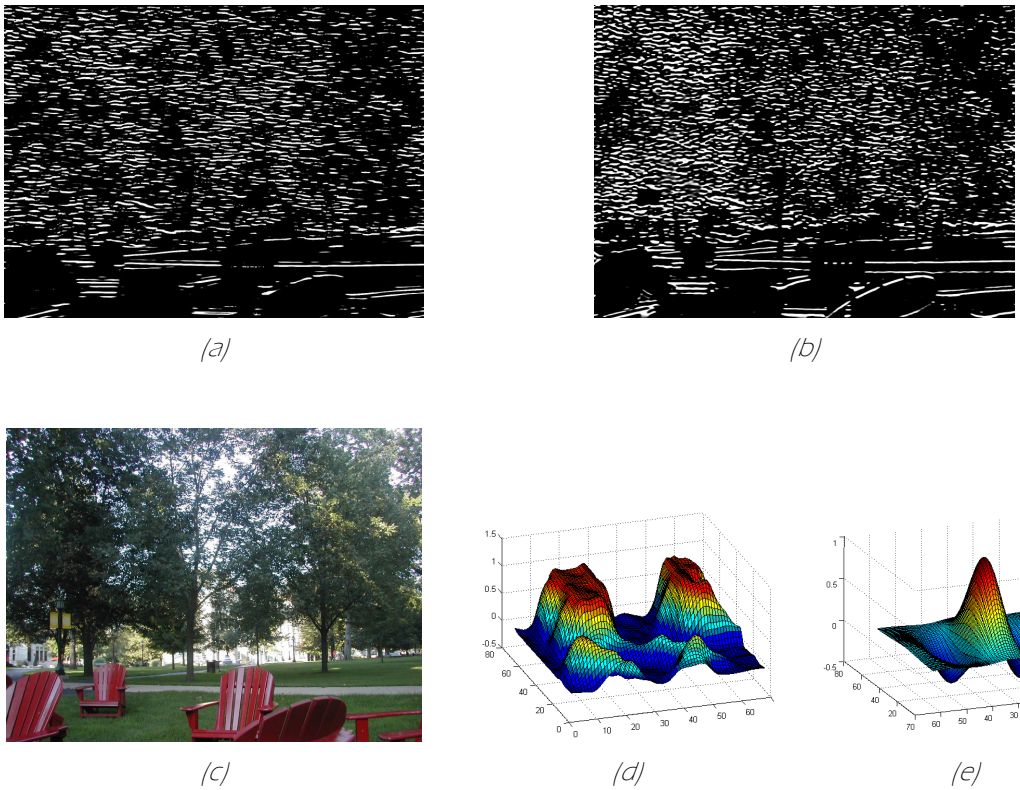
Figure 16: The complete Still Saliency Pathway. Parallelograms represent data realizations and rectangles represent processing blocks.

Many key points of this system were originally derived by Laurent Itti. There have been some simplifications and some omissions from the entire system as originally described, due to computational limitations and, otherwise, simply because the performance of the current implementation provides adequate results for the next stage of interaction.

4.2.2 Results Obtained From the Still Saliency Pathway

Although the Still Saliency Pathway is a component of a larger overall system, the results themselves are interesting and can be useful when analyzed. This section will present and comment on some results of this pathway.

Due primarily to the success of the work performed by Itti [8] and Vaingankar [18], their bottom up saliency models served as a baseline comparison for evaluating the effectiveness of the learned filters. During the filter selection process the output saliency maps of the learned ICA filters were compared to the saliency maps generated by the mathematically defined Gabor filters to determine what "type" of filter ICA had generated. This was done primarily for the orientation filters, as the color difference filters did not have distinct differences in their output saliency maps.



(a) *(b)*

(c) *(d)* *(e)*

Figure 16: Comparison of using a Learned ICA Filter (d) convolved with an image versus a Mathematically Generated Difference of Gaussian Filter (e). Original Image (c). Intensity Inverted ICA Saliency Map (a). Difference of Gaussian Saliency Map (b).

In *Figure 16* both saliency maps have very similar highlighted features. This comparison of the saliency maps created with a Learned ICA Filter (d) and a Difference of Gaussian Filter (e) that both respond highly to horizontal edges. This result is representative of the comparison to other orientations. The ICA Filters usually generate several receptive fields, which exhibit very similar features to the Difference of Gaussian Filters. An interesting feature is that they are usually generated in pairs. Although (a) is an inversion of the original intensity map, another filter generated via ICA was able to create similar results. Its output is not included because it was not used in the system implementation.

The color difference filters all seemed to produce similar output color maps, although some filters have drastically different structure to them. As stated before, there was no preference as to which color receptors were chosen. *Figure 17* shows how the color channels respond to the same image used to create the orientation maps in *Figure 16*. Some of the concepts to keep in mind are that each color filter is convolved with its respective color channel. Since yellow is not part of the RGB format and is actually composed to green and red, it does pick up some of the green in the grass. One should note the brightness of the sunspots that hit the trees and the yellow flags as compared to the grass. The chairs are obviously highlighted in the R – G color difference channel, the grass and trees in the G – R, and the sky along with the sidewalk and some of the specular highlights on the chairs and in some of the background.

(a)

(b)

(c)

(d)

(e)

Figure 17: The response maps generated by applying 17x17 Learned ICA Filters to the corresponding color channels. (a) Original Image. (b) R – G Color Difference Response. (c) G – R Color Difference Response. (d) B – Y Color Difference Response. (e) Y – B Color Difference Response.

The previous results are typical of applying the Learned ICA filters to outdoor images of the scale, which the filter patches were generated from. The picture of the red chairs was not included in the training set. It was used to determine if features from other images could produce basis functions which would bring out salient features.

Although the model implemented for summing the multiple saliency maps has been called naïve by Itti [8], and it does not have a high plausibility as the system used by the mammalian visual system, this work makes due with those facts and goes on to produce interesting and promising results. These results are focused toward generating scan paths similar to the human visual system.

(a) (b)

(c) (d)

Figure 18: Simple summations of feature saliency maps. (a)Original Image. (b)All Color Difference Maps Summed Together. (c)All the Orientation Maps Summed Together. (d)Complete Saliency Map (Color Summed With Orientation). Redder colors are more salient the features, blue is less salient. The crosswalk button is most salient.

This results obtained in *Figure 18* are again the application of the Learned ICA Filters to a given image that was not in the training data. The multiple scale multiple feature orientation map does a very good job at picking out interesting edge features, which are present in the image. The color map finds the yellow and blue channels of particular interest as can be observed in (b). When both maps are combined together through a simple summation and normalization, the final response map (d) is realized.

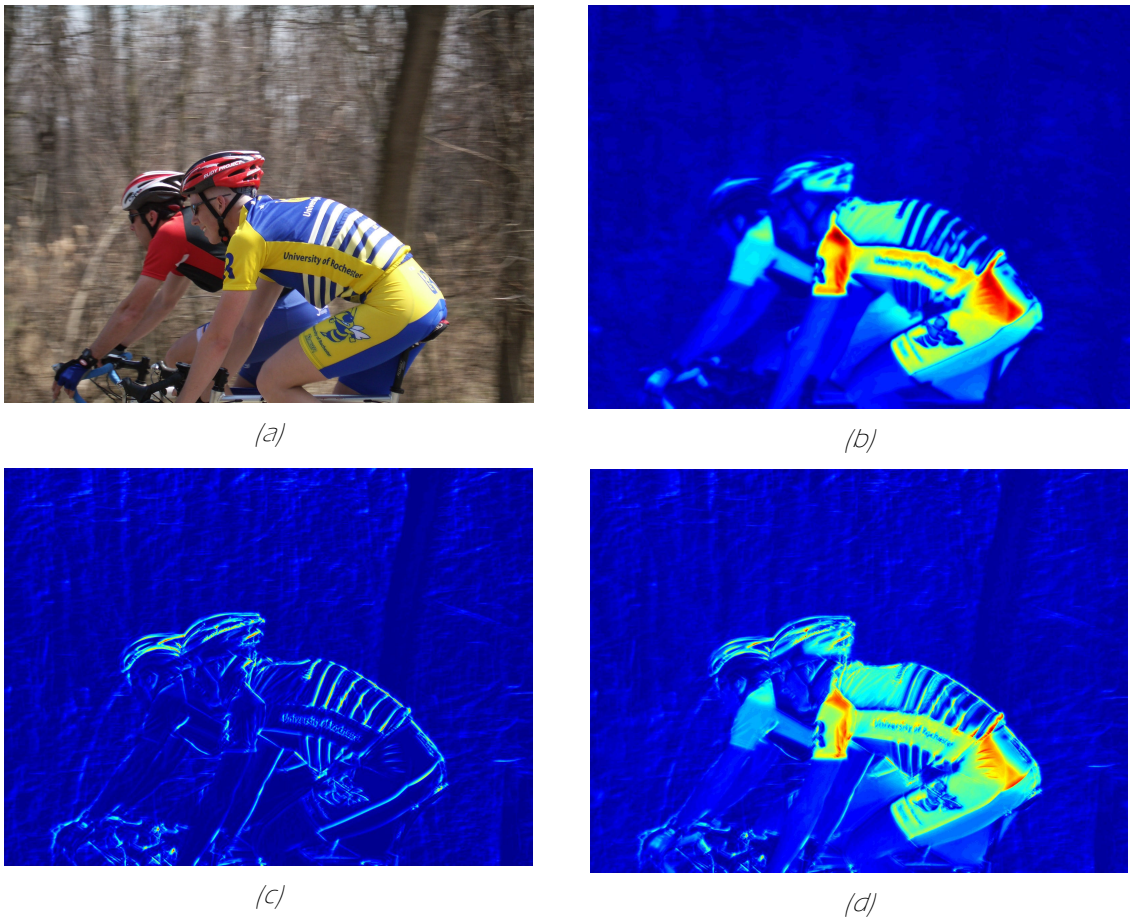


Figure 19: Another Example of Simple Summation of Saliency Maps. (a)Original Image. (b)Color Saliency Map. (c)Orientation Saliency Map. (d)Complete Saliency Map (Color Summed With Orientation).

Figure 19 is another example of an image which was sent through the Still Saliency Pathway. The interesting points to note are how much more intense the color map

(b) is as compared to the complete saliency map (d). This is due to using the orientation saliency map to help be more selective. Also, the orientation saliency map itself picks out interested features, such at the striping on the biker in the foreground. The features of both bikers' helmets is highly salient in the orientation map, but only vaguely highlighted in the color map (only due to the R-G channel responding to it). This is a good example that demonstrates the interoperability of both the color and orientation maps to provide meaningful data.

Since video is essentially a sequence of images in some given order changing at a fixed rate, the obvious extension of this pathway is to pull each frame out of a video process it, and create a new video based off of the saliency map generated. *Figure 20* is just one frame pulled out of a processed video. Notice how the colored signs are of more interest than the person in the scene.

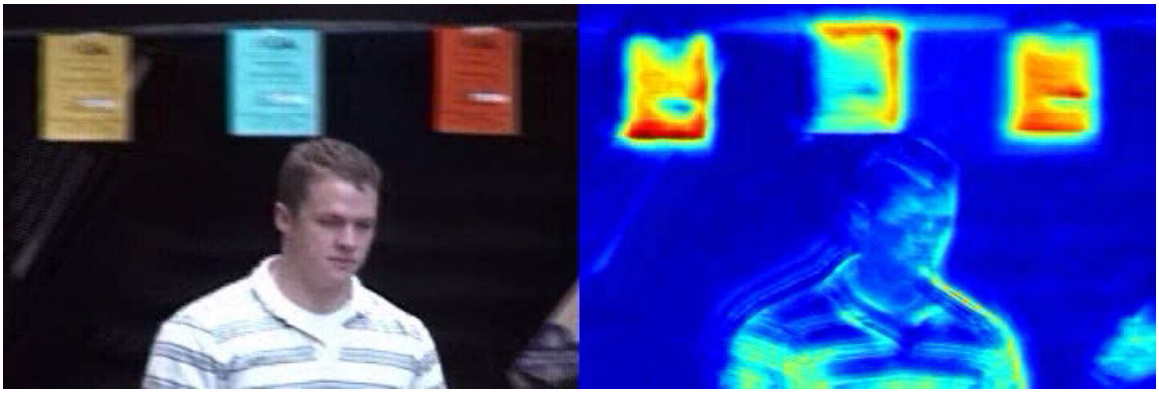


Figure 20: A single frame from a video (of different resolution than previous pictures). The left is source data and the right is the actual processed video frame for comparison. Notice that the colored signs are the most salient features in this frame of video.

4.3 Motion Processing and Top-Down Guidance

Up to this point the discussion of the implementation has only focused on the still saliency of the entire system. This still saliency pathway is designed to mimic the human's early visual system. The dorsal pathway is believed to process motion

information, which has not yet been taken into account. This next section goes into a framework for modeling the dorsal pathway of the visual cortex. Although some of the algorithms and techniques used within this framework are not biologically conceivable, the framework is flexible enough to add components that more closely model the mammalian visual pathway if so desired.

4.3.1 Motion Detection

There are many different ways to perform motion detection. Some motion detection algorithms take a biologically modeled approach and use multiple directional spatiotemporal receptors to determine motion based on the response of each filter [4]. Other algorithms are computationally inexpensive, as they use simple gradient differencing of a handful of frames [14]. The purpose of the motion detection in this particular framework is to find where events occur and build up statistical information of these events. Saliency in the case of video is typically motion. To determine novelty, motion must be detected and information gathered about an object's movement and some of its basic features (color for instance).

The motion segmentation algorithm for this work uses a more general hybrid approach than used in the surveillance system implemented by Collins *et al.* [3]. Rather than restricting the learning rate of background and the threshold rate with the same value, they are independent, and represented as α and β. The motion segmentation algorithm is governed by the following equation set:

(1) $$B_0 = I_0$$
$$T_0 \neq 0$$

(2) $$B_{n+1}(x) = \begin{cases} \beta \cdot B_n(x) + (1-\beta) \cdot I_n(x), x \in non-moving \\ B_n(x), x \in moving \end{cases}$$

(3) $$T_{n+1}(x) = \begin{cases} \alpha \cdot T_n(x) + (1-\alpha) \cdot \left(5 \times \left|I_n(x) - B_n(x)\right|\right), x \in non-moving \\ T_n(x), x \in moving \end{cases}$$

(4) $$moving \equiv \left(\left|I_n(x) - I_{n-1}(x)\right| > T_n(x)\right) \wedge \left(\left|I_n(x) - I_{n-2}(x)\right| > T_n(x)\right)$$

(5) $$blobFill \equiv \left|I_n(x) - B_n(x)\right| > T_n(x) = 1$$

Equation Set 14: A summary of the equations used for motion detection and motion segmentation. (1)The Initial Conditions, B_0 is initialized to the first frame of the video, I_0. T_0 must be initialized to a uniform non-zero matrix the size of the Image I. (2)Background model governing equation, β represents a learning rate. (3)Threshold governing equation, α represents a learning rate independent of β. (4)Motion/moving pixels are defined by this equation. (5)This equation is used to fill in the outlines.

The original work for this motion detection/segmentation algorithm, restricted both the background learning rate and the threshold learning rate to the same value, whereas this implementation makes the model more general by allowing these values to be adjusted independently of one another. Also the original work only bound the blob filling equation to bounded rectangles around the outline produced by the threshold. Due to Matlab's efficient matrix manipulation techniques, the original algorithm has been extended to work on the entire motion map, rather than just the elements within a bounding box. Some of the final extensions and improvements to Collins' *et al.* work are some simple morphological processing prior to performing the final connected component analysis, and as a pre-processing step, Gaussian smoothing.

Figure 21: A frame extracted from the motion segmentation algorithm. (upper left)Source Frame. (upper right)Motion mask applied to source. (lower left)Motion bounding box superimposed on source frame.

While the visualization of the output of this algorithm is not necessary, it aids in the explanation of the information used in the system. The information which is segmented from the background with a mask (in the upper right) is used to build up the color features of the pool of Gaussians used in this work. The red box drawn in the lower left of *Figure 21* is used in the motion tracking algorithm of this system; therefore this information used to model the motion of the video segment. This information will be described in greater detail in section 4.3.2.

4.3.2 Motion Tracking

The system currently uses a very simple object tracking routine to determine the motion of the objects in the scene. The motion is measured in the cardinal directions: North, South, East, and West with respect to the viewer of the image. The object tracking implemented is very rudimentary. It relies highly on the motion segmentation algorithm and a solid mask outline, which creates the coordinates of the bounding rectangles in the lower left of *Figure 21.* The tracking also works only for moving objects (since it is highly dependent on the motion segmentation). Once an object has stopped moving, the tracking will no longer be able to fixate on that particular object.

Tracking is accomplished through a very simple almost brute force approach. The first step in the process is to generate frame by frame output of the motion segmentation's bounding box values. The number of boxes within a frame is the number of moving objects within that particular scene. This information is stored in the *motionList*, which enables the tracking algorithm to operate primarily on the points in the list rather than directly working with the bounding values.

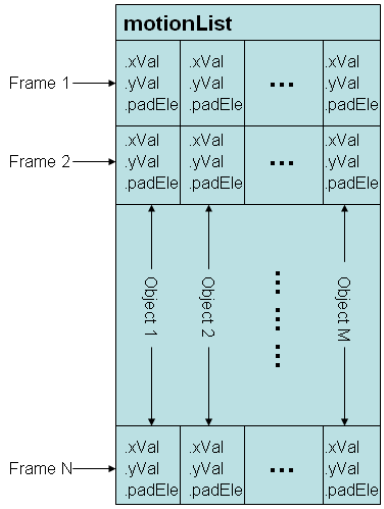


Figure 22: An example motionList. Contains N Rows and is M Columns wide, where M is the largest number of objects in any frame.

The *motionList* is very simple. Each row in the *motionList* represents a frame and each column entry in the row contains coordinates and a Boolean value representing whether the value added is for padding. Padding is added due to the fact that the width of the *motionList* is as wide as the frame that has the most motion elements. To clarify this, take for instance a sequence that only has one moving object. If at some point, due to camera jitter, a frame contains 100 objects, the motion list will end up 100 elements wide.

The motion tracking algorithm takes the current frame and the next frame and determines based on a Euclidian distance metric where the object may have moved. The method then finds the closest Euclidian distance between the two frames. It will continually do this until all the bounding rectangles have been paired together.

Due to the simplicity of this algorithm there are a few interesting cases that arise. The obvious case is when a frame of video has N bounding boxes and a new motion object is detected in the next frame. Since the current frame has N moving objects and the next frame has at least $N+1$ moving objects, a problem occurs that these bounding areas have no one to pair off with. In this implementation of the motion segmentation algorithm, this is completely ignored, therefore it will simply pair with the closest bounding box to it. Currently this does not pose a great problem since after one or two frames the tracking fixes itself, and goes to tracking the correct objects. Related in a similar fashion, is if the motion segmentation algorithm is poor and at random frames breaks a solid object into two disjoint objects (two bounding boxes). This will force the system to possibly pair up two boxes that used to be one box. The best case for this would be for the two boxes to stay disjoint for the remainder of the video, or to merge back together in the frame after the next frame. The final problem is with crossing paths. At some point the boxes will merge together and will almost snap to one another, giving a false sense of velocity. Also,

only one of the boxes will have data recorded because one box has to pair up with another. The system only requires frame to frame tracking, so these problem are not very serious.

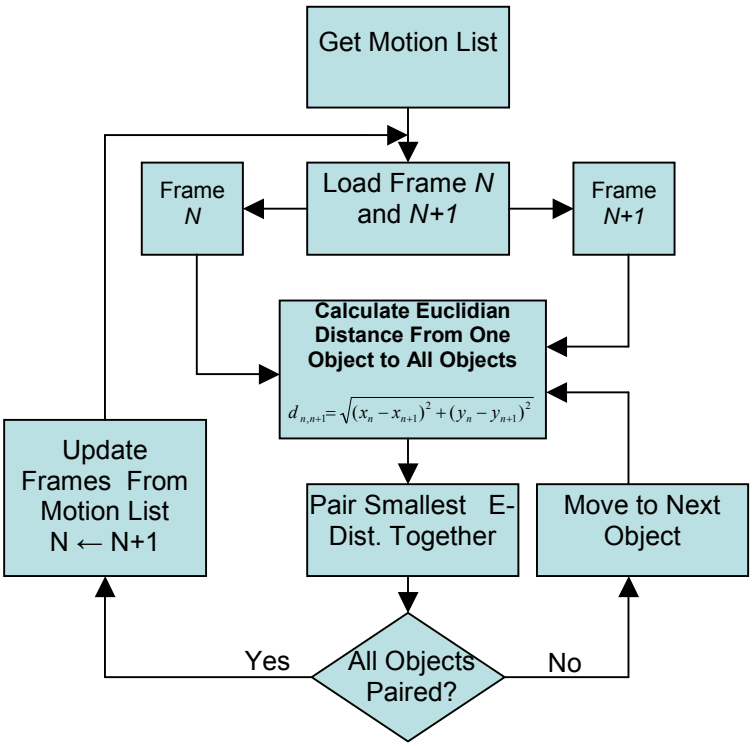


Figure 23: Flow diagram representing the motion-tracking algorithm implemented.

The saving grace of the inaccuracies of this tracking system is how much data is being gathered. Data is simply being merged into a defined statistical distribution. If there are very few data points for a certain distribution, a very low likelihood will be attached to that particular distribution, and once enough samples are measured, the overall framework will operate on those distributions.

4.4 Probabilistic Novelty Detection

The components of this system discussed thus far all have an underlying theme of detecting features within videos that generally stand out. An important component remotely associated with the visual system is the ability to become adjusted or

adapted to a certain visual stimulus. When addressing still processing, so long as the fixation of one's view changes very little (as in the case of a stationary security camera), the channel simply generates background models to compare against. To create this sort of learning/forgetting, the algorithm has a time based learning/forgetting rate.

While the saliency of still scenes can be useful, especially for modeling unconscious predefined behavior, a majority of information is detected through motion. It is beneficial to analyze what type of motion is occurring. When analyzing a stream of video, this is just the comparison of two consecutive frames. This analysis can be as simple as detecting what direction of motion occurs most in a certain region. When a region in one's field of vision detects motion in a place where motion had not previously occurred, this is an interesting event. This is similar to how people become accustomed to the motion of an oscillating fan after being exposed to it for a period of time. The back and forth motion is no longer interesting and most people can completely ignore this activity.

The remainder of this section will discuss and demonstrate the implementation of a system designed to detect novel events in a video. Key features of the system regarding the algorithmic development and testing will be presented. Final results of this component will also be included.

4.4.1 Novelty Detection Implementation & Algorithms

Motion novelty in this particular framework is given for a uniform set of patches, which subdivide the particular image. Each patch contains information regarding the range of colors that appear in that particular square, as well as information about the motion, which occurs in the given area. Although there is a wealth of information

that can be gathered from color data (as seen in *Figure 18*), it has not been rolled into the current implementation.

The idea for the motion novelty detection in this framework is to build a pool of Gaussian distributions that represent the type of movement that has occurred in the tiles that the image has been subdivided into. In this particular framework, 8x8 pixel tiles are used for the size of the quantization, though in the current implementation the size of tile quantization can be easily changed. There are four types of motion that can occur within each tile, North, South, East, and West motion. Each type of motion gets its own set of Gaussians distributions, thus information regarding the mean and standard deviation are continually updated. Updates occur when there is motion detected in that particular frame, and updates can have one of three actions.

The first action is to create a Gaussian. The first time a tile is flagged to have motion in it, the values regarding the motion (obtained from tracking) are fed in and a Gaussian is created.

The next action is a merge. Merges occur when there are fewer than 20 data points collected for a particular Gaussian curve, or when the mean of the values collected is within 3 standard deviations of any Gaussian distribution for that particular tile in that particular direction. When a merge occurs, the values that have been collected are incorporated in the Gaussian, which is closest to the mean value of the new data.

The last operation occurs if there are no Gaussian distributions in the pool for that particular motion, that are within 3 standard deviation of any of their means. This causes a split to occur which will create a new Gaussian distribution in the pool with the others. This new Gaussian follows the same rules regarding merging, and can now be used to compare against. This technique was developed in [4],

unfortunately the block diagram introduced could not be implemented as written. The implementation in this framework is as follows, and is actually adapted from [4].

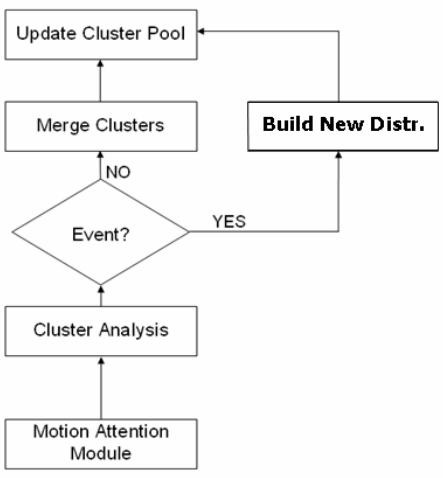


Figure 24: Split and Merge Algorithm adapted from [4]. Motion attention module is just the simple motion segmentation algorithm introduced earlier. "Event" is an outlier in data.

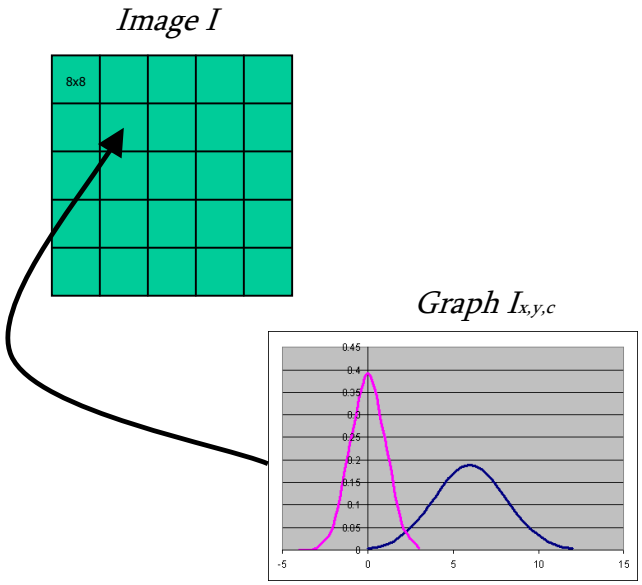


Figure 25: Representation of only one type of motion for this particular tile. Image I is 40x40 pixels but is quantized to eight tiles. Each Gaussian represents a cluster of different types of information. Every tile that has motion will build up these graphs. There can be an almost limitless amount of cluster (only restricted by physical hardware).

4.4.2 Habituation

Habituation is to become accustomed to some event through either frequent repetition or prolonged exposure. The idea of building these Gaussians is to create representations of events. The event clusters are generated, exclusively, with motion related information. Although there is no fusion between the color information and the motion information, there are functions and data structures in place that do the task of merging and splitting events.

In this system, the first time an event occurs a sharp interest is raised, which also means that a Gaussian cluster is created. The interest of this event decays, following a sigmoidal function, over a period of time, which is represented as the number of frames since the event has created the Gaussian cluster. The habituation curve for that particular event rises sharply if a similar event is encountered, which happens when a the respective cluster is merged with new data (i.e.: a similar event happened again). Given this information the habituation function is now introduced:

$$H(merges, framesPassed) = 1 - \left[\frac{1}{1 + e^{-\left(merges / framesPassed \right)}} \right]$$

Equation Set 15: *The function for habituation in this system. framesPassed is the number of frames which have passed since the cluster has been created. merges is the number of time that particular cluser has been merged with new values.*

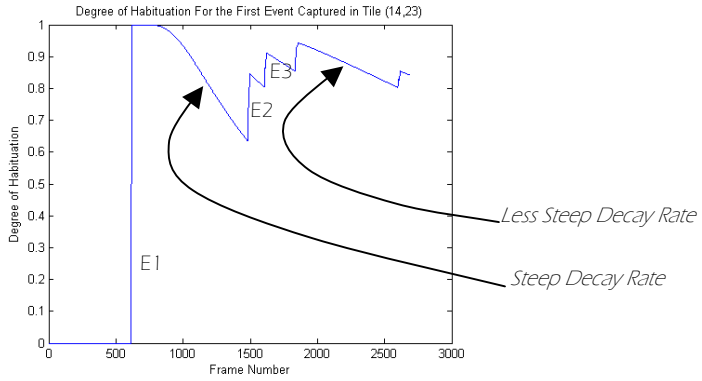
Every time a new value (or set of values) merges with a cluster, the general decaying slope becomes shallower and shallower. As this curve levels out it will eventually get to the point where there will be practically no decay, and the event will have become fully habituated. Although *Equation Set 15* is not implemented as written in [4], this equation yields results characteristic of the original VENUS system.

4.4.3 Novelty Results & Visualization

The visualization of these novelty maps is similar to how earlier figures highlight still saliency. These color maps represent the degree of habituation of a certain tile based on the first cluster created. The initial steady state of the novelty response map is set at zero. Once a response is generated this will become some non-zero value. At this first occurrence the response is quite strong and will pin the top of the scale at one. So long as no frames are merged in, this value will continually decay.

The videos that the novelty tests are performed on are typical of a security application. They are set as a stationary, non-panning camera, which is observing a moderately cluttered outdoor area. There are general pathways for people walking through the scene. The following set of images focuses primarily on the habituation which occurs for a given tile and a given cluster. To properly understand this type of information, one must actually view the interactions, which occur in the video sequence.

Figure 25 is a set of images along with a corresponding graph. The graph represents the result of applying the habituation function H to the motion list, for a given tile within the video sequence. The images below the graph represent the before and after frames of an event occurrence (look at features near the colored tile). Observe the changes that occur to this blue tile over time. This tile represents a single cluster for objects moving East in the video sequence. By looking very closely at the images on the left (before) and the image to the right (after), one will find some object that has passed through the tile. As this difference between frames occur, also note the change in color of the tile which is highlighted within the image. The graph is annotated to correspond with the movement occurrences.



Event 1 – (E1)

Frame – 580

Frame – 680

Event 2 – (E2)

Frame – 1450

Frame – 1530

Event 3 – (E3)

Frame – 1565

Frame – 1660

Figure 25: *Representation of a single tile responding to multiple people walking through at similar speeds. Notice how the tile's color changes. Redder colors imply the habituation is very high, blue tiles very low. Events always occur on rising edges. Notice how each time the event occurs, the decay rate becomes shallower. This occurs because the system is starting to learn that people walking this speed through this tile is not interesting over time.*

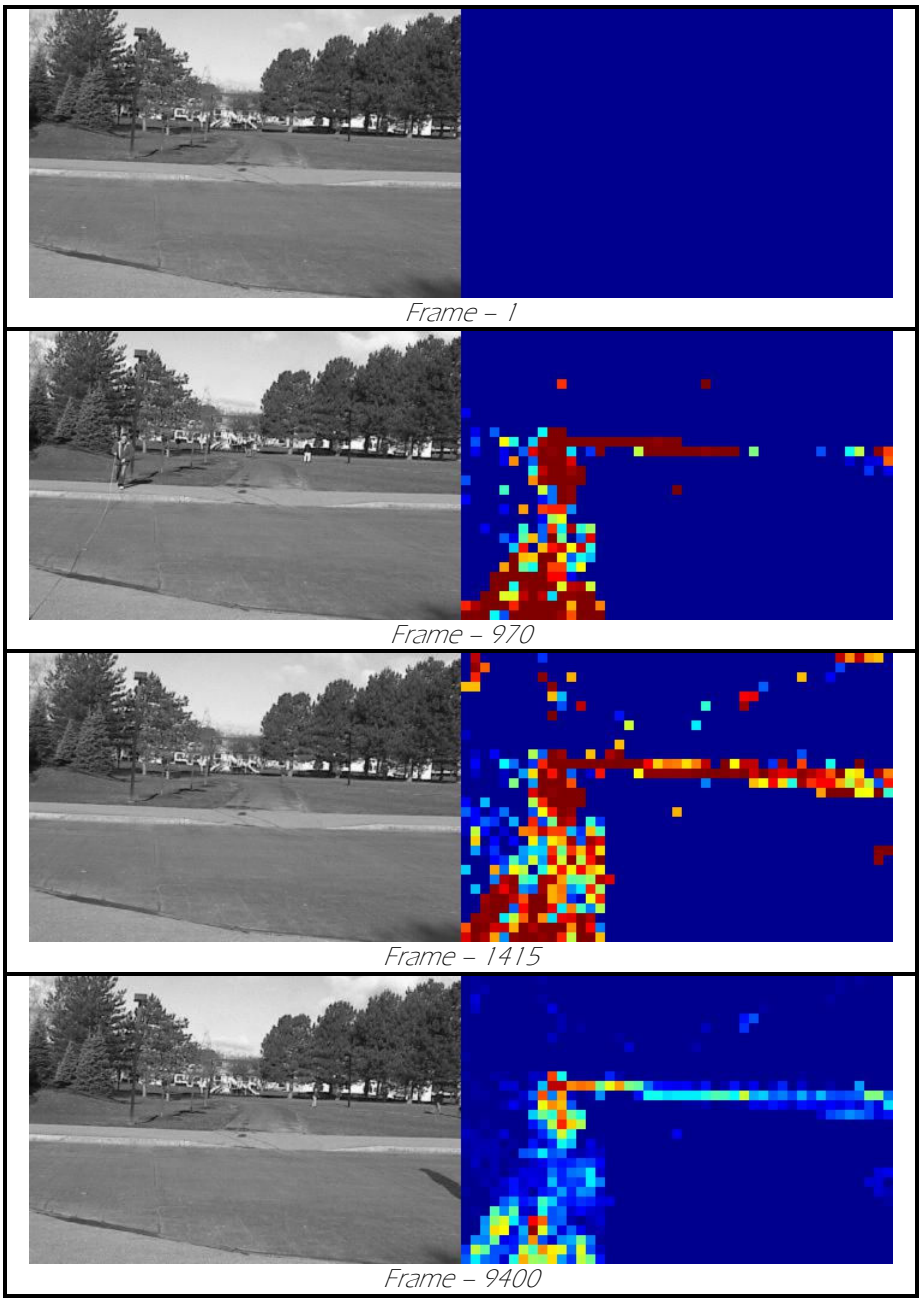


Figure 26: *A full hot map of the areas of saliency. This was taken from a video, where the left represents actual motion, and the right is the degree of habituation. The motion part of the video consists of ≈2600 Frames (near 1.5 minutes), after that, the last frame was replicated ≈6400 times to simulate no new events occurring. During the 2600 frames of motion 7 people walked through the far pathway, and two people walked back and forth to the right of the scene. Notice that at Frame 1415, there is a lot of new events occurring. Frame 9400 learns shows that most of these events decay out of the scene. The path is still habituated due to the many people walking along the pathway.*

Figure 26 is a full habituation map only visualizing the first cluster created. This is included to show how infrequent events will decay over time (the course of passing frames), whereas frequent events will have slower decay rates. Habituation is time dependent, therefore frame dependent, so it may be necessary to adjust the reinforcement rate, $\frac{merges}{famesPassed}$, by inserting a piecewise-linear multiplicative term in the denominator for different frame rates. The full novelty maps, again use the same color scheme as the still saliency maps. Colors that are dark red are highly habituated, and color that are closer to blue have a very low degree of habituation. Yellow remains in the middle of the spectrum.

V Conclusions & Future Work

The effort presented, is the framework of a system designed to detect multiple types of saliency and novelty as represented by realizing saliency maps, calculating a degree of habituation, and visualizing this degree of habituation in the form of a hot map. The framework attempts to process information as modeled by a typical human's dorsal stream and early visual pathway.

Low level receptive fields are developed over time by collecting sample patches from a large battery of images. These patches are observed and analyzed by a statistical process called Independent Component Analysis (ICA). ICA generates receptive fields, which are a set of 2D basis feature matrices. Theoretically, by combining multiple basis feature matrices in a constructive manner, intelligent information can be realized. When these feature matrices are applied to an image via convolution data is generated, which represents the amount of overlap between the feature matrix and the stimulus. By using these feature maps and naïvely summing them together, a map of key interest points is generated that seems to model how a haman would scan through the same stimulus.

Motion saliency results in the draw of attention, to a particular region in a video stream. A more specific type of motion saliency is motion detection. What becomes more interesting is determining when a particular type of motion is novel. Novelty decays over time following an exponential decay function. Once a similar event is encountered, this refreshes one's mind and the degree of habituation is increased; therefore the level of novelty is decreased. Events are described, in this system as a Gaussian distribution that is associated to a unique tile, which was quantized from the original image (in 8x8 tiles). Multiple Gaussian distributions may be associated with one tile, which would imply that two fairly different events have occurred in a particular tile.

Although positive results have been obtained from the current implementation of this system, there are several key areas that will allow for better detection of saliency and novelty in future iterations. One improvement which may be worth some research is to study different information fusion algorithms, as opposed to the naïve summation which occurs in the still saliency channel. Currently the summation's preprocessing of normalization is not very biologically plausible. In a future iteration, this has the possibility of being changed. Other features may also be incorporated into the filtering system in the still channel, such as texture, and the re-introduction of an intensity channel.

The novelty and habituation work are not at all biologically inspired. Motion detection and segmentation is generated from a simple adaptive background/threshold model. More robust methods will drastically increase the accuracy of the system. Motion tracking is also based on a very simplistic approach assuming motion occurs within very tight constraints. This can be possibly be coupled with the motion segmentation algorithm. Finally there are several extensions of this work, which would allow for more advanced novelty detection. Although the system currently has functionality, which collects and analyzes color information, this information is not currently incorporated into the novelty detection system. In future extensions, these statistics may be hooked in. Object recognition would be yet another extension. Imagine a scenario where the area of interest is a no trucks lane of a highway. Cars and trucks all move at similar speeds on the highway, so no novelty would be detected in the current iteration of this system. If the system could recognize vehicles in that specific lane as either cars or trucks, this would allow the system to flag trucks in that lane as a novel event (if truckers obey laws).

From a performance standpoint, the system should eventually be ported to a language that is meant for systems beyond the prototype stage (like C/C++). Much of

the performance hit is incurred by how Matlab works with loading large video files and writing output. This is primarily an I/O problem. The still saliency channel is currently processed serially. Since the information between all of the different response maps does not depend on one another, this problem becomes embarrassingly parallel. All of which would drastically increase performance.

Works Cited

[1] Bell, A., and Sejnowski, T. (1997). The 'Independent Components' of Natural Scenes Are Edge Filters. *Vision Research,* 37:3327-3338.

[2] Blakemore, C., and van Sluyters, R. C. (1975). Innate and Environmental Factors In the Development of the Kitten's Visual Cortex. *Journal of Physiology (London)*, 248:663–716.

[3] Collins, R., Lipton, A., Kanade, T., Fujiyoshi, H., Duggins, D., Tsin, Y., Tolliver, D., Enomoto, E., Hasegawa, O., Burt, P., Wixson, L. (2000). A System for Video Surveillance and Monitoring. *Joint Research: Carnegie Mellon University & The Sarnoff Corp. For DARPA Image Understanding & Office of Naval Research.*

[4] Gaborski, R., Vaingankar, V., Chaoji, V., Teredesai, A., Tentler, A. (2002). VENUS: A System for Novelty Detection in Video Streams With Learning. *Laboratory for Applied Computing, Rochester Institute of Technology.*

[5] Greenspan, H., Belongie, S., Goodman, R., Perona, P., Rakshit, S., Anderson, C. H. (1994). Overcomplete Steerable Pyramid Filters and Rotation Invariance. *Proc. IEEE Computer Vision and Pattern Recognition (CVPR: Seatle, WA)*

[6] Hyvärinen, A., Erkki O. (2000). Independent Component Analysis: Algorithms and Applications. *Neural Networks, 13(4-5):411-430*

[7] Hyvärinen, A.,Hoyer, P.,Hurri, J., Gutmann, M. (2005). Statistical Models of Images and Early Vision. *Proceedings of the Int. Symposium on Adaptive Knowledge Representation and Reasoning(AKRR2005).*

[8] Itti, L. (2000). Models of Bottom-Up and Top-Down Visual Attention. *Doctoral Thesis, California Institute of Technology*.

[9] Koch, C., Ullman, S. (1985). Shifts In Selective Visual Attention: Towards the Underlying Neural Circuitry. *Hum Neurobiol 4(4):219-227.*

[10] Leventhal, A.G. (1991). The Neural Basis of Visual Function. *Vision and Visual Dysfunction Volume 4. CRC Press.* Boca Raton, FL.

[11] Lindgren, J., Hyvärinen, A. (2004). Learning High-Level Independent Components of Images Through a Spectral Representation. *Proc. Int. Conf. on Pattern Recognition (ICPR2004).*

[12] Lipps, M. (2002). How People Take Pictures: Understanding Consumer Behavior through Eye Tracking Before, During, and After Image Capture. *Visual Perception Laboratory, Rochester Institute of Technology.*

[13] Liu, X., Wang, D. (2001). Appearance-Based Recognition Using Perceptual Components. *Proc. Of International Joint Conference on Neural Networks 1943-1948.*

[14] Mostefai, M., Djamila, M., Chahir, Y. (2006). Efficient Real Time Face Tracking Operator Study and Implementation within Virtex FPGA Technology. *Joint Research: University of Bordj Bou Arreridj & University of Caen.*

[15] Ohzawa, I., DeAngelis, G., Freeman, R. (1996). Encoding of Binocular Disparity by Simple Cells in the Cat's Visual Cortex. *Journal of Neurophysiology 75:1779-1805.*

[16] Pan, J., and Faloutsos, C. (2002). VideoCube: A Novel Tool For Video Mining and Classification. *Proceedings of the Fifth International Conference on Asian Digital Libraries (ICADL 2002).*

[17] Stone, J. (2004). Independent Component Analysis: A Tutorial Introduction. *MIT Press.* Cambridge Massachusetts.

[18] Vaingankar, V. (2004). Goal Directed Visual Search Based On Color Cues. *Masters Thesis, Rochester Institute of Technology.*

[19] Van Hateren, J. H., Ruderman, D. L. (1998). Independent Component Analysis of Natural Image Sequences Yields Spatio-Temporal Filters Similar to Simple Cells in Primary Visual Cortex. *Proc. R. Soc. Lond. B*, 265:2315-2320.

[20] Wang, D. L. (1995). Habituation: The Handbook of Brain Theory and Neural Networks. *MIT Press.* Cambridge Massachusetts.

[21] Weisstein, E. W. (2006). Convolution. From *MathWorld*--A Wolfram Web Resource. http://mathworld.wolfram.com/Convolution.html